



TUGAS AKHIR -SM141501

**KRIPTOGRAFI KURVA ELIPTIK DIFFIE-HELLMAN
UNTUK PROSES ENKRIPSI-DEKRIPSI CITRA DIGITAL**

DONI RUBIAGATRA
NRP 1211 100 087

DosenPembimbing
Dr. Dwi Ratna Sulitsyaningrum, S.Si, MT
Drs. Komar Baihaqi, M.Si

JURUSAN MATEMATIKA
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember
Surabaya 2017



FINAL PROJECT -SM141501

***ELLIPTIC CURVE DIFFIE-HELLMAN CRYPTOGRAPHY FOR
ENCRYPTION-DECRYPTION PROCESS OF DIGITAL IMAGE***

DONI RUBIAGATRA
NRP 1211 100 087

Supervisor
Dr. Dwi Ratna Sulitsyaningrum, S.Si, MT
Drs. Komar Baihaqi, M.Si

DEPARTMENT OF MATHEMATICS
Faculty of Mathematics and Natural Science
SepuluhNopember Institute of Technology
Surabaya 2017

LEMBAR PENGESAHAN

KRIPTOGRAFI KURVA ELIPTIK DIFFIE-HELLMAN UNTUK PROSES ENKRIPSI-DEKRIPSI CITRA DIGITAL

ELLIPTIC CURVE DIFFIE-HELLMAN CRYPTOGRAPHY FOR ENCRYPTION- DECRYPTION OF DIGITAL IMAGE

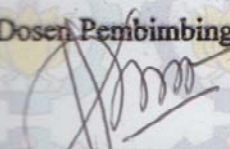
TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
Untuk memperoleh gelar Sarjana Sains
Pada bidang studi Ilmu Komputer
Program Studi S-1 Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember Surabaya


Oleh :
Doni Rubiagatra
NRP. 1211 100 087

Menyetujui,

Dosen Pembimbing II,


Drs. Komar Baihaqi, M.Si
NIP. 19600229 198803 1 001

Dosen Pembimbing I,


Dr. Dwi Ratna S. S.Si. MT
NIP. 19690405 199403 2 003

Mengetahui,

Ketua Jurusan Matematika
FMIPA ITS


Dr. Imam Mukhlash, S.Si. MT
NIP. 19700831 199403 1 003
Surabaya, Januari 2017



KRIPTOGRAFI KURVA ELIPTIK DIFFIE-HELLMAN UNTUK PROSES ENKRIPSI-DEKRIPSI CITRA DIGITAL

Nama Mahasiswa : Doni Rubiagatra
NRP : 1211 100 087
Jurusan : Matematika
Dosen Pembimbing : 1. Dr. Dwi Ratnas S, S.Si, MT
2. Drs. Komar Baihaqi, M.Si

Abstrak

Saat ini, berjuta-juta citra ditransmisikan setiap harinya melalui jaringan komputer. Jika sebuah citra mempunyai informasi penting dan pada proses transmisinya tidak dilakukan pengamanan, maka kemungkinan yang terjadi adalah adanya *intercept* dari pihak yang tidak bertanggung jawab. Citra yang di-*intercept* dapat dimanipulasi, atau informasi pada citra yang melekat sebagian dihilangkan.

Pada tugas akhir ini dibahas mengenai proses enkripsi-dekripsi citra digital untuk keamanan pesan dengan ECC (*Elliptic Curve Cryptography*). Selama ini, ECC digunakan sebagai metode yang sangat baik untuk mengamankan protokol dalam berkomunikasi di internet. Bahkan ECC dapat digunakan sebagai metode yang digunakan untuk enkripsi sebuah teks. Sehingga, diharapkan ECC dapat mengamankan citra digital dari penyadapan maupun kebocoran pesan yang bersifat rahasia. Dengan terwujudnya keamanan dalam proses pengiriman citra digital, pengirim dan penerima tidak perlu khawatir untuk berbagi citra digital yang bersifat rahasia.

Tujuan dari penelitian ini adalah untuk mengembangkan algoritma enkripsi citra digital yang tahan akan serangan. Algoritma yang diusulkan tersebut merupakan perpaduan antara ECC dengan pertukaran kunci Diffie-Hellman. Setelah melakukan enkripsi, sebuah analisis keamanan akan dilakukan pada citra cipher untuk mengevaluasi tingkat ketahanan dari teknik yang diusulkan dari sebuah serangan secara statistik.

Kata Kunci : Citra Digital, *Elliptic Curve Cryptography*, *Diffie-Hellman Key Exchange*, Enkripsi, Dekripsi

“Halaman ini sengaja dikosongkan”

**ELLIPTIC CURVE DIFFIE-HELLMAN
CRYPTOGRAPHY FOR ENCRYPTION-DECRYPTION
PROCESS OF DIGITAL IMAGE**

Name	: Doni Rubiagatra
NRP	: 1211 100 087
Department	: Mathematics
Supervisor	: 1. Dr. Dwi Ratna S, S.Si, MT 2. Drs. Komar Baihaqi, M.Si

Abstract

Nowadays, millions of images are transferred every day across the network. If an image has really important information and in the transmission, there is no security and it has a possibility of an interception. The intercepted image could be manipulated or the information in it partially removed. Cryptography is one of the solutions for securing an information.

In this undergraduate thesis, there will be an encryption-decryption process of a digital image using ECC (Elliptic Curve Cryptography). Hopefully, the digital image that has important information could be transferred securely. It means the sender of the message do not worry to share the secret information with the others.

The purpose of this research is to develop encryption algorithm that robust from several attacks, this research is a combination of ECC and Diffie-Hellman Key Exchange. After Applying encryption, security analysis is performed to evaluate the robustness of proposed technique to statistical attacks.

Keywords: *Digital Image, Elliptic Curve Cryptography, Diffie-Hellman Key Exchange, Encryption, Decryption*

“Halaman ini sengaja dikosongkan”

KATA PENGANTAR

Segala Puji bagi Allah SWT yang telah memberikan karunia, rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul: **“Kriptografi Kurva Eliptik Diffie-Hellman untuk Proses Enkripsi-Dekripsi Citra Digital”** yang merupakan salah satu persyaratan akademis dalam menyelesaikan Program Studi S-1 pada Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dapat diselesaikan dengan berkat kerjasama, bantuan, dan dukungan dari banyak pihak. Sehubungan dengan hal itu, penulis mengucapkan terima kasih kepada:

1. Dr. Dwi Ratna Sulistyaningrum, S.Si, MT dan Drs. Komar Baihaqi, M.Si selaku dosen pembimbing yang senantiasa membimbing dengan sabar dan memberikan kritik dan saran dalam penyusunan Tugas Akhir ini.
2. Dr. Imam Mukhlash, S.Si, MT selaku Ketua Jurusan Matematika ITS.
3. Prof. DR. Basuki Widodo, M.Sc selaku Dosen Wali.
4. Dr. Imam Mukhlash, S.Si, MT dan Drs. Mohammad Setijo Winarko, M.Si selaku dosen penguji Tugas Akhir ini.
5. Dr. Didik Khusnul Arif, S.Si, M.Si selaku Kaprodi Sarjana Matematika
6. Drs. Iis Herisman, M.Si selaku Sekertaris Kaprodi Sarjana Matematika
7. Seluruh jajaran dosen dan staf jurusan Matematika ITS.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan saran dan kritik dari pembaca. Akhir kata, semoga Tugas Akhir ini bermanfaat bagi semua pihak yang berkepentingan.

Surabaya, Januari 2017

Penulis

special thanks to

Selama proses pembuatan Tugas Akhir ini, banyak pihak yang telah memberikan bantuan dan dukungan untuk penulis. Penulis mengucapkan terima kasih dan apresiasi secara khusus kepada:

1. Ayah Sih Budiono dan Ibu Rusmiati tersayang yang senantiasa dengan ikhlas memberikan semangat, perhatian, kasih sayang, doa, motivasi dan nasihat yang sangat berarti bagi penulis.
2. Kakek Jatim yang telah berpulang ke Ilahi terlebih dahulu sebelum Saya wisuda dan menjadi salah satu motivasi bagi penulis untuk segera menyelesaikan Tugas Akhir ini.
3. Tahta Dari Timur yang telah meluangkan waktu untuk membantu penulis untuk mengerjakan tugas akhir ini dan menyiapkan SageMathCloud sebagai media komputasi.
4. Dimaz Wisnu Adipradana yang membantu dalam pemahaman konsep aljabar pada Tugas Akhir ini.
5. Kabinet BEM FMIPA ITS 13/14 : Alfia Puji Rahayu, Aziz Nugroho, Ummu Habibah Nur Azizah, Denni Hariyanto, Angga Firmansyah, Sos Edwin Vidiyoga, Juniarto Setyo Nugroho, Indi Yasinta Hadiani Fikliani, Husna Elsaviaristi, Keysha Wellviestu Zakri, Aprillia Tri Wahyuni Utami, Fauziah Gistri Destilunna, Fiscy Aprilia Rahmanika, Silvana Dwi Nurherdiana, Ika Restu Affianti, Nova Kusuma Putri, Nurul Alfiyah, Cordova Ulin Nuha Kamila, Gita Ayu Apriliyana, Annisa Sajidah, Taqy Thianugraha, Lilis Wahyu Astutik, Raffy Rishady Subandrio. Terima kasih atas kekeluargaan dan pengalaman berharga dalam satu kepengurusan. Mohon maaf jika saya mempunyai kesalahan saat menjadi Ketua maupun secara personal dan menjadi yang lulus terakhir dalam kabinet.

6. Teman-teman Himatika ITS 12/13 dan BEM FMIPA ITS 12/13 serta 13/14 sukses selalu.
7. Teman-teman TEDxTuguPahlawan dan surabaya.py yang merupakan tempat dimana saya bisa berdiskusi dan menjadi penyemangat selama saya menjalani ekstensi masa studi sarjana Matematika ITS.

Tentu saja masih banyak pihak lain yang turut andil dalam penyelesaian tugas akhir ini yang tidak bisa penulis sebutkan satu persatu. Semoga Allah membalas dengan balasan yang lebih baik bagi semua pihak yang telah membantu penulis. *Amin ya rabbal 'alamin.*

“Halaman ini sengaja dikosongkan”

DAFTAR ISI

	Halaman
HALAMAN JUDUL	i
LEMBAR PENGESAHAN	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR	xi
DAFTAR ISI	xv
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xix

BAB I. PENDAHULUAN

1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah	3
1.4 Tujuan	3
1.5 Manfaat.....	4
1.6 Sistematika Penulisan Tugas Akhir	4

BAB II. TINJAUAN PUSTAKA

2.1 Pengertian Citra	7
2.2 Citra Digital	8
2.2.1 Citra Biner	8
2.2.2 Citra <i>Grayscale</i>	8
2.2.3 Citra Berwarna.....	8
2.3 Digitalisasi Spasial (Sampling).....	9
2.4 Kriptografi	9
2.4.1 Algoritma Simetris.....	10
2.4.2 Algoritma Asimetris	11
2.5 <i>Diffe-Hellman Key Exchange Algorithm</i>	12
2.6 <i>Elliptic Curve Cryptography</i>	13
2.6.1 Kurva Eliptik pada Himpunan F_p	15
2.7 Domain Parameter Kurva Eliptik	17

BAB III. METODOLOGI

3.1	Objek Penelitian.....	19
3.2	Peralatan.....	19
3.3	Tahap Penelitian.....	19
BAB IV. PERANCANGAN DAN IMPLEMENTASI SISTEM		
4.1	Perancangan Kurva Eliptik di bidang F_p	23
4.1.1	Pembuatan semua titik (x, y)	23
4.1.2	Penambahan Titik pada Kurva Eliptik (x_3, y_3)	27
4.2	Representasi Piksel pada Kurva Eliptik	31
4.3	Pertukaran Kunci Diffie-Hellman	33
4.4	Perancangan Sistem Enkripsi ECDH	33
4.5	Perancangan Sistem Dekripsi ECDH.....	35
4.6	Matriks yang dibentuk dari Citra dengan Format JPG	36
4.7	Implementasi pada SageMathCloud	37
4.7.1	SageMathCloud.....	37
4.7.2	Pembacaan Piksel dalam Citra.....	39
4.7.3	Pembuatan Titik Kurva Eliptik beserta Tabel Pemetaan	40
4.7.4	Pertukaran Kunci Diffie-Hellman	43
4.7.5	Proses Pemetaan dan Enkripsi Citra	45
4.7.6	Proses Pemetaan dan Dekripsi Citra	48
BAB V. PENGUJIAN DAN PEMBAHASAN		
5.1	Pengujian Titik Kurva Eliptik	51
5.2	Pengujian Citra Hasil Enkripsi	53
5.3	Pengujian Berbagai Ukuran dan Macam Citra	57
BAB VI. KESIMPULAN DAN SARAN		
6.1	Kesimpulan	59
6.2	Saran	60
DAFTAR PUSTAKA		61
LAMPIRAN		63

DAFTAR GAMBAR

	Halaman
Gambar 2.1	Diagram proses enkripsi dan dekripsi..... 10
Gambar 2.2	Pertukaran kunci Diffie-Hellman..... 13
Gambar 3.1	Diagram alir proses metode penelitian 21
Gambar 3.2	Diagram alir proses (a) enkripsi (b) dekripsi 22
Gambar 4.1	Flowchart pembuatan titik (x,y)..... 26
Gambar 4.2	Flowchart pembuatan titik jika $P = Q$ 27
Gambar 4.3	Flowchart pembuatan titik jika $P \neq Q$ 29
Gambar 4.4	Diagram alir proses enkripsi pada citra 34
Gambar 4.5	Diagram alir proses dekripsi pada citra..... 36
Gambar 4.6	Citragray.jpg dengan ukuran 5×5 37
Gambar 4.7	Matriks citragray.jpg pada SAGE dan OpenCV 37
Gambar 4.8	Interface dari SageMathCloud notebook 38
Gambar 4.9	Tampilan nama program Python yang dibuat 38
Gambar 4.10	Lenna100.jpg citra <i>grayscale</i> berukuran 100×100 39
Gambar 4.11	Import Library dan pembacaan piksel pada citra..... 39
Gambar 4.12	Class KurvaEliptik 40
Gambar 4.13	Class Titik pada KurvaEliptik 41
Gambar 4.14	Fungsi semuaTitik 41
Gambar 4.15	Pendefinisian kurva eliptik SMC..... 42
Gambar 4.16	Pembuatan tabel pemetaan pada SMC..... 42
Gambar 4.17	Potongan hasil tabel pemetaan pada SMC..... 43
Gambar 4.18	Listing program multiplikasi skalar untuk Titik 44
Gambar 4.19	Implementasi Diffie-Hellman pada SMC 44
Gambar 4.20	Proses pemetaan setiap piksel pada SMC..... 45
Gambar 4.21	Listing program penambahan dua titik pada Kurva Eliptik 46
Gambar 4.22	Proses enkripsi pada SMC 46
Gambar 4.23	Proses pengembalian titik cipher menjadi citra kembali 47
Gambar 4.24	Hasil Citra lenna100 yang telah terenkripsi 47
Gambar 4.25	Proses dekripsi menjadi titik plain kembali pada SAGE..... 48
Gambar 4.26	Proses pengembalian titik dekripsi menjadi citra kembali 49
Gambar 4.27	Citra hasil dekripsi 49

Gambar 5.1	(a) Citra plain, (b) Citra cipher	53
Gambar 5.2	(a) Histogram citra plain, (b) Histogram citra cipher	53
Gambar 5.3	Scatter plot korelasi horizontal (a) citra plain, (b) citra cipher.....	54
Gambar 5.4	Scatter plot korelasi vertikal (a) Citra plain, (b) citra cipher.....	55
Gambar 5.5	Scatter plot korelasi diagonal (a) Citra plain, (b) Citra cipher.....	55
Gambar 5.6	(a) Citra dekripsi dengan kunci sesuai, $k =$ 2345 (b) Citra dekripsi dengan kunci tidak sesuai, $k = 2346$	56

DAFTAR TABEL

	Halaman
Tabel 4.1 Hasil QR_{17} (Quadratic Residue Module).....	25
Tabel 4.2 Elemen grup kurva eliptik.....	26
Tabel 4.3 Tabel pemetaan piksel pada titik kurva eliptik $E_{7211}(1,7206)$	32
Tabel 5.1 Pengujian titik kurva eliptik.....	51
Tabel 5.2 Nilai Korelasi Citra Plain dan Citra Cipher	54
Tabel 5.2 Hasil uji coba berbagai jenis citra.....	57

“Halaman ini sengaja dikosongkan”

BAB I

PENDAHULUAN

1.1 Latar Belakang

Saat ini berjuta-juta citra ditransmisikan setiap hari melalui jaringan komputer. Jika sebuah citra mempunyai informasi penting dan pada proses transmisinya tidak dilakukan pengamanan, maka kemungkinan yang terjadi adalah adanya *intercept* dari pihak yang tidak bertanggung jawab. Citra yang di-*intercept* dapat dimanipulasi, atau informasi pada citra yang melekat sebagian dihilangkan.

Citra mempunyai peranan penting dalam representasi informasi dan banyak digunakan dalam berbagai aplikasi seperti dalam bidang militer, komunikasi, maupun telemedis. Biasanya, citra ditransmisikan melalui jaringan yang tidak aman. Sehingga, untuk keamanan citra dari *intercept*, duplikasi dan penghilangan informasi menjadi sebuah topik yang hangat didiskusikan oleh kalangan peneliti. Dalam citra medis, segi keamanan merupakan hal yang penting karena menyangkut pada privasi pasien. Sampai saat ini, tanggung jawab data medis para pasien dipercayakan sepenuhnya oleh petugas medis. Hal ini berarti semua data medis dari pasien tidak boleh diketahui oleh orang lain karena sifatnya yang sensitif dan rahasia. Tetapi, dengan berkembangnya teknologi komputer terutama di bidang telemedis, munculnya suatu tantangan tersendiri dalam menjaga privasi dan keamanan data medis pasien. Sehingga, kita harus memastikan citra medis yang ditransmisikan dapat divalidasi integritas dan kerahasiaannya.

Dalam perkembangannya, banyak algoritma enkripsi yang ditawarkan untuk mengamankan suatu citra. Algoritma enkripsi yang lazim antara lain dengan mengganti nilai suatu piksel dan/atau lokasinya. Algoritma enkripsi tersebut dapat diklasifikasikan menjadi enkripsi citra pada domain frekuensi dan pada domain spasial. Enkripsi pada domain frekuensi didesain

untuk mengganti data pada citra di dalam domain frekuensi, seperti *Discrete Fractional Fourier Transform*[1], *Quantum Fourier Transform*[2] dan *Reciprocal-Orthogonal Parametric Transform*[3]. Enkripsi pada domain spasial dikenal sebagai teknik substitusi-permutasi dengan mengganti nilai piksel dari suatu citra dan lokasinya. Contoh dari proses substitusi-permutasi adalah enkripsi *P-Fibonacci Transform* [4], *Wave Transmission*[5], *Elliptic Curve Cryptography*[6], dan *Chaotic System*[7]. Kedua algoritma tersebut menghasilkan hasil enkripsi citra dengan keamanan yang tinggi. Namun, kedua algoritma tersebut menghasilkan terbatas hanya dua macam jenis luaran yaitu *noise-like* dan *texture-like*[8] .

Elliptic Curve Cryptography atau yang lebih dikenal dengan ECC, merupakan metode kriptografi yang penggunaannya semakin pesat setiap tahunnya dan mulai menggantikan metode RSA. Metode RSA pada awal perkenalannya merupakan metode yang sangat kuat. Bahkan, sang penemu RSA, *Rivest-Shamir-Adleman* mengutarakan bahwa untuk membobol sistem ini adalah dengan memecahkan permasalahan matematika yang sangat sulit.. Mencari sebuah faktor adalah permasalahan yang dimaksud dan hal ini sudah ada dari sejak jaman dahulu. Jika ada sebuah cara untuk mencari sebuah faktor, maka penemuan ini akan menjadi sebuah berita yang sangat besar. Karena hampir semua sistem kriptografi saat ini menggunakan metode RSA, maka akan terjadi perombakan besar besaran jika masalah tersebut terpecahkan. Dalam sistem bit, terdapat sebuah algoritma untuk mencari faktor faktor tersebut. Seiring dengan berkembangnya jaman, sumber daya komputer saat ini cukup untuk melakukan perhitungan dalam memecahkan algoritma RSA. Satu satunya cara yang ditawarkan untuk menanggulangiya adalah dengan menambahkan kunci bit agar semakin besar pada RSA. Hal ini berakibat sistem RSA bukan merupakan sistem yang ideal untuk masa depan kriptografi. Sehingga belakangan ini, ECC mulai mengambil alih peranan RSA dalam sistem kriptografi yang ada.

Berdasarkan latar belakang serta kajian pada beberapa penelitian tersebut, kami ingin mengembangkan sebuah solusi dalam mengamankan pengiriman pesan berupa citra digital menggunakan *Elliptic Curve Cryptography (ECC)*. Algoritma enkripsi-dekripsi citra digital yang ditawarkan dan menjadi sebuah judul tugas akhir yaitu Kriptografi Kurva Eliptik Diffie-Hellman untuk Proses Enkripsi-Dekripsi Citra Digital.

1.2 Rumusan Masalah

Rumusan masalah dari Tugas Akhir ini adalah :

1. Bagaimana melakukan proses enkripsi dan dekripsi citra digital menggunakan kriptografi kurva eliptik Diffie-Hellman..
2. Bagaimana hasil kinerja metode kriptografi kurva eliptik Diffie-Hellman pada citra digital, sehingga dapat diamankan dari serangan secara statistik.

1.3 Batasan Masalah

Batasan Masalah yang akan dibahas dalam penelitian tugas akhir ini adalah sebagai berikut :

1. Citra yang digunakan adalah citra Grayscale dalam format JPG, JPEG
2. Bahasa Pemrograman yang digunakan adalah Python dan diaplikasikan pada SageMathCloud
3. Pertukaran Kunci yang digunakan adalah Diffie-Hellman(DH)

1.4 Tujuan

Tujuan dari tugas akhir ini adalah :

1. Melakukan proses enkripsi dan dekripsi citra digital menggunakan kurva eliptik

2. Memperlihatkan hasil kinerja kinerja metode kriptografi kurva eliptik Diffie-Hellman pada citra digital, sehingga dapat diamankan dari serangan secara statistik.

1.5 Manfaat

Manfaat dari tugas akhir ini adalah :

1. Bagi institusi medis, militer dll, hasil penelitian ini diharapkan dapat dijadikan salah satu alternatif untuk menjaga keamanan dan privasi dari citra digital.
2. Bagi institusi pendidikan, diharapkan dapat dijadikan bahan pengetahuan untuk penelitian selanjutnya yang lebih mendalam berkaitan dengan enkripsi citra digital
3. Bagi masyarakat , diharapkan tidak khawatir terhadap privasi dan keamanan dari citra digital.

1.6 Sistematika Penulisan Tugas Akhir

Sistematika penulisan didalam Tugas Akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang pembuatan tugas akhir, rumusan dan batasan permasalahan yang dihadapi dalam penelitian tugas akhir, tujuan dan manfaat pembuatan tugas akhir dan sistematika penulisan tugas akhir.

BAB II KAJIAN TEORI

Bab ini menjelaskan tentang penelitian sebelumnya yang mengkaji metode enkripsi citra digital. Selain itu, pada bab ini akan dijelaskan kajian teori dari referensi penunjang serta penjelasan permasalahan yang dibahas dalam tugas akhir ini, meliputi Pengertian Citra Digital, Digitalisasi spasial, Kriptografi, Algoritma Simetris, Algoritma Asimetris, *Diffie-Hellman Key-Exchange Algorithm*, dan *Elliptic Curve Cryptography*.

BAB III METODE PENELITIAN

Bab ini berisi metodologi atau urutan pengerjaan yang dilakukan dalam menyelesaikan tugas akhir, meliputi studi literatur, analisis dan desain kriptosistem, pembuatan program, uji coba dan evaluasi program, penarikan kesimpulan, penulisan laporan tugas akhir, dan seminar.

BAB IV PERANCANGAN DAN IMPLEMENTASI

Bab ini menjelaskan analisis desain kriptosistem, Pada tahapan ini akan dilakukan analisis citra dan parameter-parameter yang dibutuhkan dalam pembuatan desain program. Sistem ini memiliki dua proses utama yaitu enkripsi dan dekripsi. Perancangan sistem juga terdiri dari perancangan kurva eliptik di bidang F_p , pembuatan semua titik, penambahan titik pada kurva eliptik, representasi piksel pada kurva eliptik, pertukaran kunci Diffie-Hellman, perancangan sistem ECDH, matriks yang dibentuk dari citra, dan Implementasi pada SageMathCloud

BAB V PENGUJIAN DAN PEMBAHASAN HASIL

Pada tahap ini akan dilakukan pengujian terhadap citra yang telah dienkripsi dan sistem yang telah dibangun. Diantaranya adalah pengujian titik kurva eliptik dan pengujian citra hasil enkripsi

BAB VI PENUTUP

Bab ini merupakan penutup, berisi tentang kesimpulan yang dapat diambil berdasarkan data yang ada dan saran yang selanjutnya dilakukan bila tugas akhir ini dilanjutkan.

“Halaman sengaja dikosongkan”

BAB II

TINJAUAN PUSTAKA

Pada bab ini menjelaskan tentang penelitian sebelumnya yang mengkaji metode enkripsi citra digital. Selain itu, pada bab ini akan dijelaskan kajian teori dari referensi penunjang serta penjelasan permasalahan yang dibahas dalam tugas akhir ini, meliputi Pengertian Citra Digital, Digitalisasi spasial, Kriptografi, Algoritma Simetris, Algoritma Asimetris, *Diffie-Hellman Key-Exchange Algorithm*, dan *Elliptic Curve Cryptography*.

2.1 Pengertian Citra

Citra menurut Kamus Besar Bahasa Indonesia memiliki makna rupa, gambar, atau gambaran. Sedangkan menurut kamus Webster citra adalah suatu representasi, kemiripan, atau imitasi dari suatu objek atau benda. Citra terbagi menjadi dua yaitu citra diam dan citra bergerak. Citra diam adalah citra tunggal yang tidak bergerak. Sedangkan, citra bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun sehingga memberi kesan pada mata kita sebagai gambar yang bergerak.

Dalam beberapa masa, citra yang dikenal manusia berbentuk citra kontinu. Suatu representasi objek yang dihasilkan dari sistem optik yang menerima sinyal analog dan dinyatakan dalam bidang dua dimensi. Nilai cahaya yang ditransmisikan pada citra kontinu memiliki rentang nilai yang tak terbatas. Contoh dari citra kontinu adalah mata manusia dan kamera analog. Sebuah citra analog tidak dapat direpresentasikan secara langsung oleh komputer. Oleh sebab itu dilakukan sebuah proses untuk merubah nilai-nilai yang ada pada citra analog agar komputer dapat membaca dan menerjemahkan informasi yang terdapat pada citra analog. Hasil dari pemrosesan tersebut dinamakan sebagai citra digital.

2.2 Citra Digital

Citra digital merupakan fungsi dua dimensi yang dapat dinyatakan dengan fungsi $f(x,y)$, dimana x dan y merupakan titik koordinat spasial. Dan amplitudo dari fungsi f pada sembarang koordinat (x,y) merupakan nilai intensitas cahaya, yang merupakan representasi dari warna cahaya yang ada pada citra analog. Citra digital adalah suatu citra dimana (x,y) dan nilai intensitas dari f terbatas (discrete quantities), dan telah dilakukan proses digitalisasi spasial dan digitalisasi kuantitas[9].

2.2.1 Citra Biner

Citra biner memiliki 2 macam warna yaitu hitam dan putih. Citra biner membutuhkan 1 bit untuk menyimpan dua warna ini. Nilai untuk warna berupa angka 0 untuk hitam dan 1 untuk warna putih[9].

2.2.2 Citra *Grayscale*

Berbeda dengan citra biner yang hanya memiliki warna hitam dan putih, citra *grayscale* memiliki data dalam bentuk matriks dengan representasi tingkat keabuan $[0,255]$. [9]

2.2.3 Citra Berwarna

Setiap piksel dalam citra berwarna memiliki tiga nilai yang diambil dari tiga nilai dasar Red, Green Blue atau RGB. Setiap dari warna dasar menggunakan penyimpanan 8 bit dengan gradasi $[0,255]$. Sehingga setiap piksel mempunyai kombinasi warna sebanyak 16 juta warna lebih. Hal ini menyebabkan format citra berwarna hampir menyerupai dari citra aslinya.[9]

2.3 Digitalisasi Spasial (Sampling)

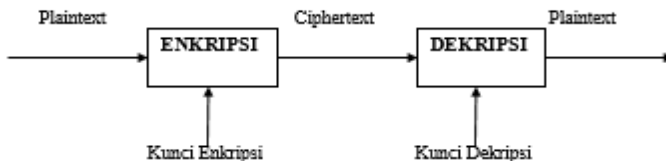
Sampling merupakan proses pengambilan informasi dari citra analog yang memiliki panjang dan lebar tertentu untuk membaginya ke beberapa blok kecil. Blok-blok tersebut disebut sebagai piksel. Sehingga citra digital yang lazim dinyatakan

dalam bentuk matriks memiliki ukuran $M \times N$ dengan M sebagai baris dan N kolom. Bisa juga disebut sebagai citra digital yang memiliki $M \times N$ buah piksel. Notasi matriks citra digital dapat dinyatakan sebagai persamaan 2.1 berikut[9]:

$$f(x, y) = \begin{bmatrix} f(0,0) & \dots & f(0, N-1) \\ \vdots & \ddots & \vdots \\ f(M-1, 0) & \dots & f(M-1, N-1) \end{bmatrix} \quad (2.1)$$

2.4 Kriptografi

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan suatu informasi. Dalam kriptografi terdapat dua konsep utama yaitu proses enkripsi dan dekripsi. Enkripsi adalah proses dimana informasi atau data yang hendak dikirim diubah menjadi bentuk yang tidak dapat dikenali oleh orang awam. Dekripsi adalah kebalikan dari enkripsi yaitu mengubah bentuk yang tidak dikenali menjadi informasi awal[10].



Gambar. 2.1 Diagram proses enkripsi dan dekripsi

Dapat dilihat pada gambar 2.1 bahwa masukan berupa *plaintext* akan masuk ke dalam blok enkripsi dan luarannya berupa *chipertext*. Kemudian *chipertext* akan masuk ke dalam blok dekripsi dan keluarannya berupa *plaintext*.

Kriptografi modern menyelesaikan masalah enkripsi dan dekripsi dengan merahasiakan kunci saja tanpa harus merahasiakan algoritmanya. Kunci ini merupakan nilai yang sangat spesifik dan bekerja dengan algoritma kriptografi untuk menghasilkan pesan yang terenkripsi secara spesifik pula. Dengan kunci inilah nantinya kita akan dapat melakukan proses enkripsi dan dekripsi. Karena keamanan bergantung pada kerahasiaan kuncinya, maka algoritma yang dibentuk dapat dianalisa dan dipublikasikan .

Berdasarkan jenis kunci yang digunakannya, algoritma kriptografi dikelompokkan menjadi dua bagian, yakni algoritma simetris dan algoritma asimetris.

2.4.1 Algoritma Simetris

Algoritma simetris merupakan merupakan kriptografi konvensional dengan menggunakan satu kunci enkripsi. Algoritma ini sangat luas digunakan sebelum berkembangnya algoritma asimetris pada tahun 1970. Algoritma yang memakai kunci simetris diantaranya adalah [10] :

1. *Data Encryption Standard* (DES) yakni algoritma yang tergolong jenis blok kode. DES beroperasi pada ukuran blok 64 bit. DES mengenkripsi 64 bit teks-asli menjadi 64 bit teks-kode dengan menggunakan 56 bit *internal key* atau *subkey*. Kunci internal dibangkitkan dari kunci eksternal yang panjangnya 64 bit.
2. *Advance Encryption Standard* (AES) yakni algoritma yang memiliki 3 blok cipher yaitu AES-128, AES-192, AES-256 yang diadopsi dari koleksi yang lebih besar yang awalnya diterbitkan sebagai Rijndael. Masing-masing cipher memiliki ukuran 128 bit dengan ukuran kunci masing-masing 128, 192, dan 256 bit.
3. *International Data Encryption Standard* (IDEA) yakni algoritma yang menggunakan konfusi dan difusi. Dari kunci yang mempunyai panjang 128 bit dibangkitkan 52

subkey. Algoritma IDEA menggunakan 52 *subkey* dan 16 bit kunci per blok.

4. A5 yakni suatu aliran kode yang digunakan untuk mengamankan percakapan telepon selular GSM. Aliran kodenya terdiri dari 3 buah *Linear Feedback Shift Register* (LSFR) yang dikontrol oleh blok dengan LSFR 19 bit, 22 bit, dan 23 bit. Masing-masing dari LSFR memiliki periode berturut-turut dan .
5. *One Time Pad* (OTP) yakni algoritma yang berisi deretan kunci yang dibangkitkan secara acak. Setiap kunci hanya digunakan untuk sekali pakai. Pemilihan kunci harus secara acak agar tidak bisa diproduksi ulang dan membuat lawan tidak mudah menerka. Jumlah karakter kunci sama dengan jumlah karakter yang dimiliki pesan.
6. RC2, RC4, RC5, RC6 dan lainnya.

2.4.2 Algoritma Asimetris

Algoritma Asimetris adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsinya. Pada algoritma ini, proses enkripsinya menggunakan kunci publik dimana kunci tersebut tidak perlu dijaga kerahasiaannya. Dan untuk proses dekripsinya menggunakan kunci privat dan bersifat rahasia. Algoritma yang memakai kunci asimetris diantaranya adalah [10]:

1. *Digital Signature Algorithm* (DSA) yakni algoritma yang menggunakan kunci umum untuk membuktikan pesan yang diterima sama dengan identitas pengirim data. DSA mempunyai penghitungan yang sulit karena pemisahan algoritma dan berorientasi pada algoritma.ElGamal dan Schnorr.
2. RSA yakni algoritma yang melakukan pemfaktoran bilangan yang sangat besar. Untuk membangkitkan dua kunci, dipilih dua bilangan prima acak yang besar. RSA mengekspresikan teks asli yang dienkripsi menjadi blok-blok yang mana setiap blok memiliki nilai bilangan biner yang diberi symbol “n”, blok teks asli “M” dan blok teks

kode “C”. Untuk melakukan enkripsi pesan “M”, pesan dibagi ke dalam blok-blok numerik yang lebih kecil daripada “n” (data biner dengan pangkat terbesar). Jika bilangan prima yang panjangnya 200 digit, dapat ditambah beberapa bit 0 di kiri bilangan untuk menjaga agar pesan tetap kurang dari nilai “n”.

3. Diffie-Hellman (DH) yakni algoritma yang memiliki keamanannya dari kesulitan menghitung logaritma diskrit dalam *finite field*, dibandingkan kemudahan dalam menghitung bentuk eksponensial dalam *finite field* yang sama. Algoritma ini dapat digunakan dalam mendistribusikan kunci public yang dikenal dengan protokol pertukaran kunci
4. *Elliptic Curve Cryptography* (ECC) yakni algoritma yang mendasarkan keamanannya pada permasalahan matematis kurva eliptik. Tidak seperti permasalahan matematis logaritma diskrit dan pemfaktoran bilangan bulat, tidak ada algoritma waktu sub-10 eksponensial yang diketahui memecahkan permasalahan matematis algoritma kurva eliptik.

2.5 Diffie-Hellman Key Exchange Algorithm

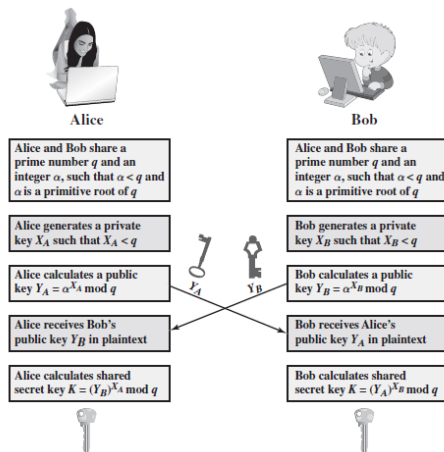
Algoritma Diffie Hellman bergantung kepada kesulitannya memecahkan masalah logaritma diskrit. Sebuah akar primitif dari bilangan prima p adalah pangkat modulo p yang menggenerasi semua bilangan bulat dari 1 sampai $p - 1$. Misalkan a adalah sebuah akar primitif dari p maka [10]:

$$a \bmod p, a^2 \bmod p, a^{p-1} \bmod p \quad (2.2)$$

Pada persamaan 2.2, untuk setiap bilangan bulat b dan sebuah akar primitif a dan bilangan prima p kita dapatkan eksponen i sehingga.

$$b \equiv a^i(\text{mod } p), \quad 0 \leq i \leq (p - 1) \quad (2.3)$$

Eksponen i dapat dikatan sebuah logaritma diskrit b untuk basis $a, \text{mod } p$ seperti pada persamaan 2.3. Kemudian akan digambarkan algoritma dari pertukaran kunci Diffie-Hellman.



Gambar 2.2 Pertukaran kunci Diffie-Hellman

2.6 Elliptic Curve Cryptography

Elliptic Curve Cryptosystem (ECC) diperkenalkan tahun 1985 oleh Neal Koblitz dan Victor Miller dari Universitas Washington. Kurva eliptik mempunyai masalah logaritma yang terpisah sehingga sulit untuk dipecahkan. Pada Juni 2000 kunci enkripsi ECC yang memakai 108 bit (yang setara dengan kunci enkripsi RSA 600 bit), berhasil dipecahkan menggunakan 9500 komputer yang berjalan paralel selama 4 bulan yang dihubungkan dengan internet.

Kriptografi kurva eliptik termasuk sistem kriptografi kunci publik yang mendasarkan keamanannya pada permasalahan matematis kurva eliptik. Tidak seperti permasalahan matematis logaritmis diskrit (*Discrete Logarithm Problem*, DLP) dan pemfaktoran bilangan bulat (*Integer Factorization Problem*, IFP), tidak ada algoritma waktu sub-eksponensial yang diketahui untuk memecahkan permasalahan matematis algoritma diskrit kurva eliptik (*Elliptic Curve Discrete Logarithm Problem*, ECDLP). Oleh karena alasan tersebut algoritma kurva eliptik mempunyai keuntungan bila dibanding algoritma kriptografi kunci publik lainnya, yaitu dalam hal ukuran kunci yang lebih pendek tetapi memiliki tingkat keamanan yang sama.

Kurva eliptik yang digunakan dalam kriptografi didefinisikan dengan menggunakan dua tipe daerah terbatas yakni daerah karakteristik ganjil (F_p dimana $p > 3$ adalah bilangan prima yang besar) dan karakteristik dua (F_{2^m}). Karena perbedaan itu menjadi tidak begitu penting, kedua daerah terbatas tersebut dapat ditunjukkan sebagai F_p , dimana $q = p$ atau $q = 2^m$. Elemen dari F_p adalah integer ($0 \leq x < p$) dimana elemen tersebut dapat dikombinasikan menggunakan modul aritmatik.

Pada bagian ini akan dibahas teknik dasar kurva eliptik dalam bidang terbatas F_p dimana p adalah bilangan prima lebih besar dari 3. Selanjutnya kurva eliptik secara umum didefinisikan sebagai *field* berhingga (*finite field*). Sebuah kurva eliptik E didefinisikan dalam persamaan 2.4[11]:

$$y^2 = x^3 + ax + b \quad (2.4)$$

dimana $a, b \in F_p$ dan $4a^3 + 27b^2 \neq 0$ dan sebuah titik \mathbf{O} yang disebut titik tak hingga (*infinity*). Titik tak hingga adalah

identitas atau titik ideal. Himpunan $E(F_p)$ adalah semua titik (x, y) untuk $x, y \in F_p$ yang memenuhi persamaan 2.4 .

Untuk menjelaskan uraian di atas, berikut ini diberikan contoh pencarian himpunan pada \mathbb{R} dan $E(F_p)$. Diberikan persamaan kurva eliptik $E: y^2 = x^3 + x + 1$. Untuk $E(F_p)$ dipilih $p = 23$, sehingga grup $F_{23}(a = 1, b = 1)$. Maka untuk nilai $4a^3 + 27b^2 = 4 + 27 \neq 0$ membuat E ada dalam kurva eliptik.

2.6.1 Kurva Eliptik pada Himpunan F_p

Pada bidang terbatas F_p perhitungan dilakukan dengan menggunakan aturan-aturan aritmatika modular. Persamaan kurva eliptik pada F_p dapat dituliskan sebagai berikut :

$$y^2 \equiv (x^3 + ax + b) \text{mod } p \quad a, b \in F_p \quad (2.5)$$

dengan p adalah p bilangan prima ganjil. $F_p(a, b)$ adalah himpunan yang terdiri atas titik-titik yang memenuhi persamaan 2.5 ditambah dengan titik \mathbf{O} yang disebut titik *infinity*. Kurva eliptik pada bidang terbatas merupakan grup abelian, apabila sisi kanan persamaan 2.5 tidak memiliki faktor yang berulang yaitu apabila koefisien-koefisiennya memenuhi persamaan $a^3 + b^2 \text{mod } p \neq 0$. Operasi yang berlaku dalam bidang terbatas adalah:

1. Penjumlahan (*addition*), jika $a, b \in F_p$, maka $a + b = r$ dimana r adalah sisa pembagian $a + b$ dengan bilangan prima p , $0 \leq r \leq p - 1$. Penjumlahan seperti ini disebut penjumlahan modulo p ($\text{mod } p$)

2. Perkalian (*multiplication*) jika $a, b \in F_p$, maka $a, b = s$, dimana s adalah sisa pembagian a, b dengan bilangan prima p , $0 \leq s \leq p - 1$. Perkalian seperti ini perkalian modulo p ($\text{mod } p$).

Penjumlahan dua buah titik $P(x_1, y_1)$ dan $Q(x_2, y_2)$ adalah (x_3, y_3) dengan syarat bahwa $P \neq \mathbf{O}$ dan $Q \neq P$. Secara aljabar, (x_3, y_3) diperoleh dengan rumus berikut:

$$\lambda = \left(\frac{y_2 - y_1}{x_2 - x_1} \right) \text{ mod } p \quad \text{Jika } P \neq Q \quad (2.6)$$

$$\lambda = \left(\frac{3x_1^2 - a}{2y_1} \right) \text{ mod } p \quad \text{Jika } P = Q \quad (2.7)$$

$$x_3 = \lambda^2 - x_1 - x_2 \text{ mod } p \quad (2.8)$$

$$y_3 = -y_1 + \lambda(x_1 - x_3) \text{ mod } p \quad (2.9)$$

Operasi penjumlahan pada kurva eliptik atas F_p didefinisikan sebagai berikut:

- \mathbf{O} adalah identitas penjumlahan, sehingga $P + \mathbf{O} = \mathbf{O} + P = P$ untuk setiap $P \in E(F_p)$
- Jika $P = (x, y)$ maka $P + (x, -y) = \mathbf{O}$. Titik $(x, -y)$ adalah negatif P atau $(-P)$
- Misalkan $P = (x_1, y_1) \in E(F_p)$ dan titik $Q = (x_2, y_2) \in E(F_p)$ dimana $P \neq \mathbf{O}$, $Q \neq \mathbf{O}$ dan $Q \neq \pm P$. maka $P + Q = (x_3, y_3)$ dimana :

$$x_3 = \left[\left(\frac{y_2 - y_1}{x_2 - x_1} \right)^2 - x_1 - x_2 \right] \text{ mod } p \quad (2.10)$$

$$y_3 = \left[-y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1} \right) (x_1 - x_3) \right] \text{ mod } p \quad (2.11)$$

- d. Misalkan $P = (x_1, y_1) \in E(F_p)$ maka $P + P = 2P = (x_3, y_3)$ dimana:

$$x_3 = \left[\left(\frac{3x_1^2 - a}{2y_1} \right)^2 - 2x_1 \right] \bmod p \quad (2.12)$$

$$y_3 = \left[-y_1 + \left(\frac{3x_1^2 - a}{2y_1} \right) (x_1 - x_3) \right] \bmod p \quad (2.13)$$

2.7 Domain Parameter Kurva Eliptik

Sebelum mengimplementasikan sebuah kriptografi kurva eliptik, dipersiapkan parameter yang dibutuhkan oleh sistem kriptografi tersebut. Sehingga seluruh pengguna sistem dapat mengetahui beberapa parameter yang akan digunakan bersama. Parameter ini bersifat umum dan boleh diketahui oleh setiap pengguna sistem tersebut.

Pembuatan domain parameter tersebut tidak dilakukan oleh masing-masing pengirim atau penerima karena akan melibatkan perhitungan jumlah titik pada kurva yang akan memakan waktu yang lama dan sulit untuk diterapkan. Domain parameter kurva eliptik atas F_p didefinisikan sebagai persamaan 2.14 :

$$T = (p, a, b, G, n, h) \quad (2.14)$$

Dimana

- p : bilangan prima
- a, b : koefisien persamaan kurva eliptik
- G : titik dasar (base point) yaitu elemen pembangun grup kurva eliptik
- n : order dari G yaitu bilangan bulat positif terkecil $\ni n.G = O$
- h : kofaktor, $h = \# \frac{E}{n}$, $\#E$ adalah jumlah titik dalam grup eliptik $E_p(a, b)$

Halaman sengaja dikosongkan”

BAB III

METODE PENELITIAN

Bab ini membahas mengenai metodologi sistem yang digunakan untuk menyelesaikan tugas akhir. Pembahasan metodologi sistem diawali dengan penjelasan tentang objek penelitian, peralatan yang digunakan, dan tahap penelitian.

3.1 Objek Penelitian

Objek penelitian yang akan digunakan pada tugas akhir adalah citra *grayscale* dari standard image test.

3.2 Peralatan

Peralatan penelitian yang digunakan untuk menyelesaikan tugas akhir ini adalah SageMathCloud sebagai perangkat lunak utama untuk mengaplikasikan bahasa pemrograman Python.

3.3 Tahap Penelitian

Adapun tahap-tahap yang dilakukan dalam penyusunan tugas akhir ini dan dapat dilihat pada gambar 3.1 adalah sebagai berikut :

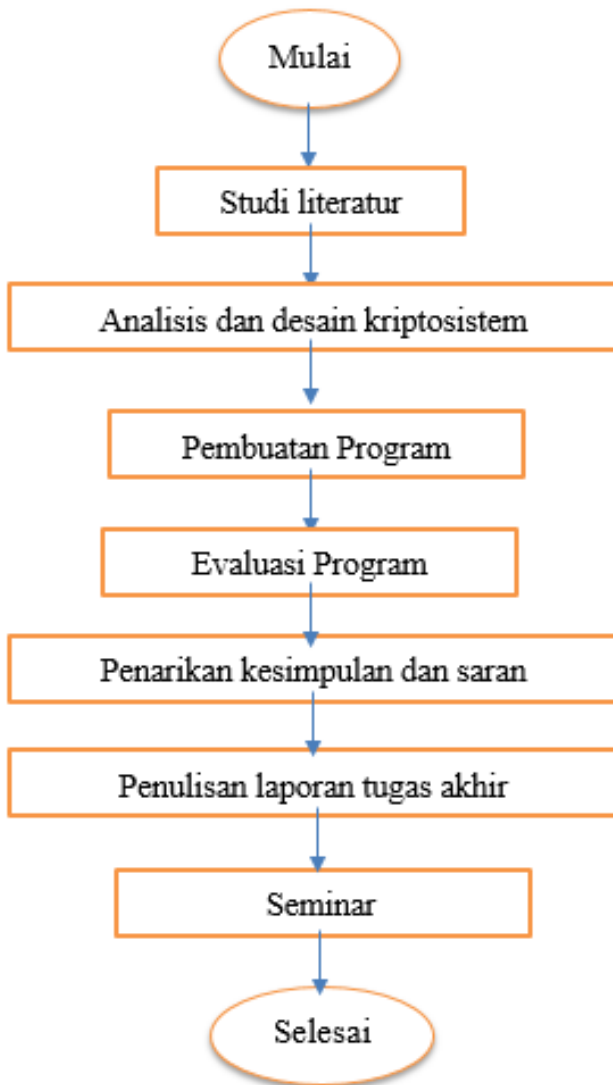
1. Studi literatur tentang *Elliptic Curve Cryptography*

Pada tahap ini akan dikaji tentang penggunaan dari enkripsi citra digital. Studi ini dilakukan dengan membaca jurnal *Elliptic Curve Cryptography*, berdiskusi dengan dosen, dan melakukan tanya jawab bersama alumni terkait konsep-konsep tersebut.

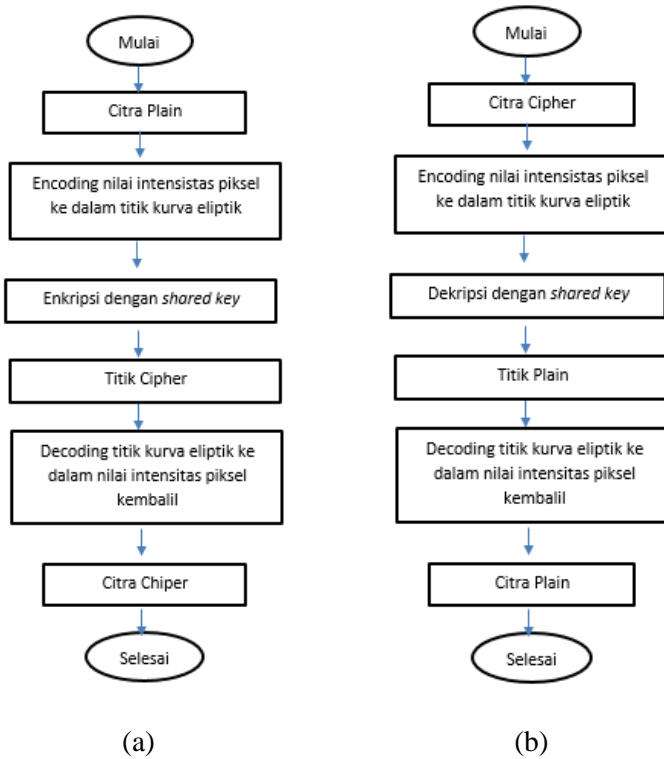
2. Analisis dan desain kriptosistem

Pada tahapan ini akan dilakukan analisis citra dan parameter-parameter yang dibutuhkan dalam pembuatan desain program. Sistem ini memiliki dua proses yaitu enkripsi dan dekripsi yang dapat dilihat pada gambar 3. 2.

3. Pembuatan program
Pada tahap ini akan dilakukan pembuatan program sesuai hasil analisis dan desain sistem menggunakan bahasa pemrograman Python dan diaplikasikan ke dalam aplikasi SageMathCloud.
4. Uji coba dan evaluasi program
Pada tahap empat ini akan dilakukan pengujian terhadap citra yang telah dienkripsi dan sistem yang telah dibangun. Untuk mengetahui ketahanan pada citra akan dilakukan analisis dan serangan secara statistik melalui uji coba diagram, korelasi, entropi, dan sensitifitas kunci.
5. Penarikan kesimpulan
Tahap kelima ini akan ditarik kesimpulan dari hasil pengerjaan, uji coba, dan pembahasan serta disampaikan saran untuk pengembangan berikutnya.
6. Penulisan laporan tugas akhir
Tahap keenam ini merupakan tahap yang terakhir. Penulis akan menuliskan semua hasil yang telah didapatkan selama penelitian ini.
7. Seminar
Tahap terakhir yang akan dilakukan adalah seminar. Pada tahap ini akan didaftarkan artikel ilmiah yang telah dibuat kepada seminar nasional maupun internasional



Gambar. 3.1 Diagram alir metode penelitian



Gambar. 3.2 Diagram alir proses (a) Enkripsi (b) Dekripsi

BAB IV

PERANCANGAN DAN IMPLEMENTASI SISTEM

Pada bab ini dibahas mengenai perancangan dan implementasi dari sistem EC-Diffie Hellman. Perancangan sistem meliputi perancangan kurva eliptik di bidang F_p , domain parameter, pembangkitan kunci publik dan kunci privat, perancangan sistem enkripsi dan dekripsi citra. Implementasi sistem meliputi pembuatan program secara keseluruhan dengan menggunakan SageMathCloud dan Python.

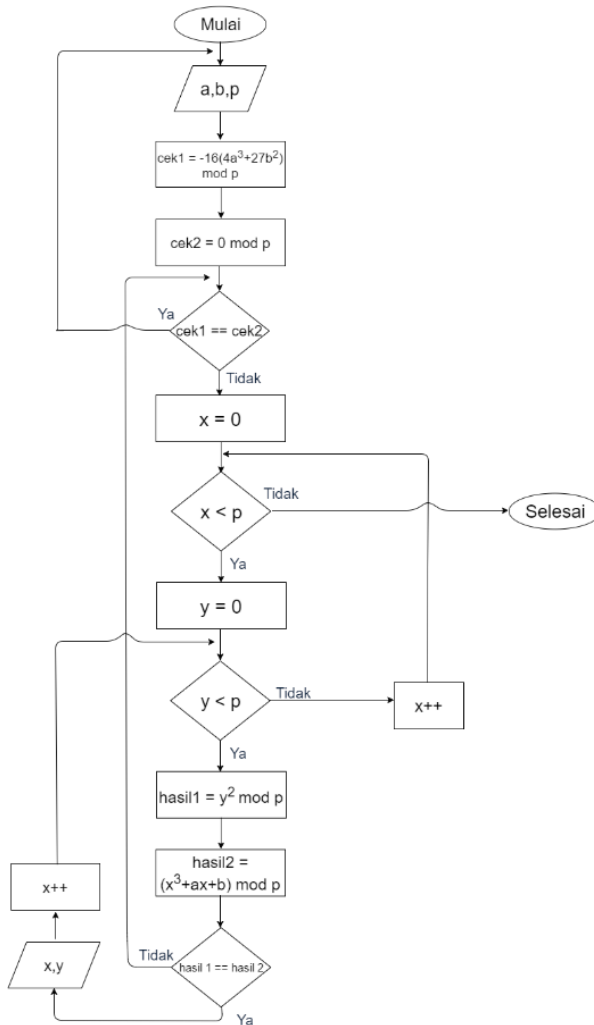
4.1 Perancangan Kurva Eliptik di bidang F_p

Sistem kriptografi tidak menggunakan kurva eliptik pada bilangan real namun menggunakan medan terbatas misalnya medan modular bilangan prima F_p . Persamaan kurva eliptik pada F_p dapat dituliskan seperti persamaan 2.5 dengan p adalah bilangan prima ganjil dengan $p > 3$.

4.1.1 Pembuatan semua titik (x, y)

Pada gambar 4.1 dijelaskan bahwa nilai a dan $b \in F_p$. Kemudian nilai a dan b diproses sesuai dengan persamaan $a^3 + b^2 \pmod{p} \neq 0 \pmod{p}$, jika tidak memenuhi persamaan tersebut maka harus kembali ke awal untuk memasukkan nilai a, b dan p . Jika memenuhi persamaan tersebut maka proses akan berlanjut untuk menemukan titik (x, y) yakni dengan syarat $x < p$, jika tidak memenuhi maka berhenti dari proses, dan jika memenuhi maka berlanjut untuk syarat $y < p$. Jika syarat tersebut tidak terpenuhi, maka berlanjut untuk syarat $x++$ dan kembali ke awal syarat $x < p$. Jika syarat tersebut terpenuhi maka akan menuju ke proses selanjutnya yakni hasil $1 = y^2 \pmod{p}$ dan hasil $2 = (x^3 + ax + b) \pmod{p}$. Jika hasil 1 \neq hasil 2 maka proses akan kembali pada pengecekan persamaan $a^3 + b^2 \pmod{p} \neq 0 \pmod{p}$, jika hasil 1 = hasil 2 maka akan

ditemukan titik (x, y) dan proses tersebut berulang ($y++$) terus sampai syarat tidak terpenuhi.



Gambar 4.1 Flowchart pembuatan titik (x, y)

Contoh perhitungan secara aljabar untuk pembuatan titik kurva eliptik, diberikan persamaan kurva eliptik $E: y^2 = x^3 + x + 5$ dengan $p = 17$, yaitu grup $F_{17}(a = 1, b = 5)$. Maka untuk nilai $4a^3 + 27b^2 = 4(1) + 27(25) \neq 0$, sehingga E ada dalam kurva eliptik.

Untuk dapat membuat titik kurva (x, y) , pertama tentukan elemen dari kurva eliptik atas $E_{17}(1, 5)$ atas F_p :

$$F_{17} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$$

Sebelum menentukan derah elemen kurva eliptik QR_{17} (Quadratic Residue Module).

Tabel 4.1 Hasil QR_{17} (Quadratic Residue Module)

F_p	$y^2(\text{mod } 17)$	QR_{17}
0	$0^2(\text{mod } 17)$	0
1	$1^2(\text{mod } 17)$	1
2	$2^2(\text{mod } 17)$	4
3	$3^2(\text{mod } 17)$	9
4	$4^2(\text{mod } 17)$	16
5	$5^2(\text{mod } 17)$	8
6	$6^2(\text{mod } 17)$	2
7	$7^2(\text{mod } 17)$	15
8	$8^2(\text{mod } 17)$	13
9	$9^2(\text{mod } 17)$	13
10	$10^2(\text{mod } 17)$	15
11	$11^2(\text{mod } 17)$	2
12	$12^2(\text{mod } 17)$	8
13	$13^2(\text{mod } 17)$	16
14	$14^2(\text{mod } 17)$	9
15	$15^2(\text{mod } 17)$	4
16	$16^2(\text{mod } 17)$	1

Jadi didapat $QR_{17} = \{0, 1, 2, 4, 8, 9, 13, 15, 16\}$

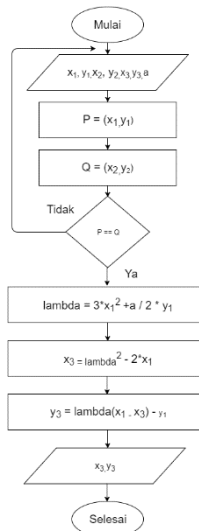
Menentukan elemen grup kurva eliptik $E_{17}(1, 5)$ yang merupakan himpunan penyelesaian dari $y^2 = x^3 + x + 5 \pmod{17}$ untuk $x \in F_{17}$ dan $y^2 = QR_{17}$

Tabel 4.2 Elemen grup kurva eliptik

$x \in F_{17}$	$y^2 = x^3 + x + 5 \pmod{17}$	$y^2 \in QR_{17}$	$(x, y) \in E_{17}(1, 5)$
$x = 0$	$y^2 = 0^3 + 0 + 5 \pmod{17}$ $= 5$	$5 \notin QR_{17}$	-
$x = 1$	$y^2 = 1^3 + 1 + 5 \pmod{17}$ $= 7$	$7 \notin QR_{17}$	-
$x = 2$	$y^2 = 2^3 + 2 + 5 \pmod{17}$ $= 15$	$15 \in QR_{17}$	(2,7) dan (2,10)
$x = 3$	$y^2 = 3^3 + 3 + 5 \pmod{17}$ $= 35 \pmod{17} = 1$	$1 \in QR_{17}$	(3,1) dan (3,16)
$x = 4$	$y^2 = 4^3 + 4 + 5 \pmod{17}$ $= 73 \pmod{17} = 5$	$5 \notin QR_{17}$	-
$x = 5$	$y^2 = 5^3 + 5 + 5 \pmod{17}$ $= 135 \pmod{17} = 16$	$16 \in QR_{17}$	(5,4) dan (5,13)
$x = 6$	$y^2 = 6^3 + 6 + 5 \pmod{17}$ $= 227 \pmod{17} = 6$	$6 \notin QR_{17}$	-
$x = 7$	$y^2 = 7^3 + 7 + 5 \pmod{17}$ $= 355 \pmod{17} = 15$	$15 \in QR_{17}$	(7,7) dan (7,10)
$x = 8$	$y^2 = 8^3 + 8 + 5 \pmod{17}$ $= 525 \pmod{17} = 15$	$15 \in QR_{17}$	(5,4) dan (5,13)
$x = 9$	$y^2 = 9^3 + 9 + 5 \pmod{17}$ $= 743 \pmod{17} = 12$	$12 \notin QR_{17}$	-
$x = 10$	$y^2 = 10^3 + 10 + 5 \pmod{17}$ $= 1015 \pmod{17} = 12$	$12 \notin QR_{17}$	-
$x = 11$	$y^2 = 11^3 + 11 + 5 \pmod{17}$ $= 1347 \pmod{17} = 4$	$4 \in QR_{17}$	(11,2) dan (11,13)
$x = 12$	$y^2 = 12^3 + 12 + 5 \pmod{17}$ $= 1745 \pmod{17} = 11$	$11 \notin QR_{17}$	-
$x = 13$	$y^2 = 13^3 + 13 + 5 \pmod{17}$ $= 2215 \pmod{17} = 5$	$5 \notin QR_{17}$	-
$x = 14$	$y^2 = 14^3 + 14 + 5 \pmod{17}$ $= 2763 \pmod{17} = 9$	$9 \in QR_{17}$	(14,3) dan (14,14)
$x = 15$	$y^2 = 15^3 + 15 + 5 \pmod{17}$ $= 3395 \pmod{17} = 12$	$12 \notin QR_{17}$	-
$x = 16$	$y^2 = 16^3 + 16 + 5 \pmod{17}$ $= 4117 \pmod{17} = 3$	$3 \notin QR_{17}$	-

4.1.2 Penambahan Titik pada Kurva Eliptik (x_3, y_3)

Pada gambar 4.2 menjelaskan proses pembuatan titik ketiga di bidang terbatas f_p dengan penjumlahan dua titik yang sama. Pada proses pembuatan titik ketiga ini terdapat dua titik yang sama yakni $P = (x_1, y_1)$ dan $Q = (x_2, y_2)$, untuk nilai $x_1, y_1, x_2, y_2, x_3, y_3 \in E(F_p)$. Syarat pertama untuk proses ini adalah $P = Q$, jika syarat tersebut tidak terpenuhi maka kembali ke awal untuk x_1, y_1, x_2, y_2 jika syarat terpenuhi maka ke proses selanjutnya yakni melakukan penghitungan menggunakan rumus lamda pada persamaan 2.7. Setelah nilai lamda ditemukan, kemudian x_3 dapat dihitung dengan menggunakan persamaan 2.8. Selanjutnya y_3 dapat dihitung dengan menggunakan persamaan 2.9. Titik ketiga x_3, y_3 telah ditemukan dari proses diatas dan diagram alir proses pembuatan titik ketiga dengan penjumlahan dua titik yang sama dapat dilihat pada gambar 4.2.



Gambar 4.2 Flowchart penambahan Titik jika $P = Q$

Contoh perhitungan secara aljabar untuk pembuatan titik ketiga dengan titik awal sama, untuk persamaan kurva eliptik dengan didapatkan titik kurva eliptik $y^2 = x^3 + x + 5$ dengan $a = 1, b = 5, p = 17$ didapatkan titik kurva eliptik

$$(x, y) = \{(2,7), (2,10), (3,1), (5,4), (5,13), (7,7), (7,10)\} \\ \{(8,7), (8,10), (11,2), (11,15), (14,3), (14,14)\}$$

Untuk $P = (x_1, y_1)$ dan $Q = (x_2, y_2)$ maka $P = Q = P + P = 2P = (x_3, y_3)$ dimana :

$$x_3 = \left[\left(\frac{3x_1^2 + a}{2y_1} \right)^2 - 2x_1 \right] \bmod p$$

$$y_3 = \left[\left(\frac{3x_1^2 + a}{2y_1} \right) (x_1 - x_3) - y_1 \right] \bmod p$$

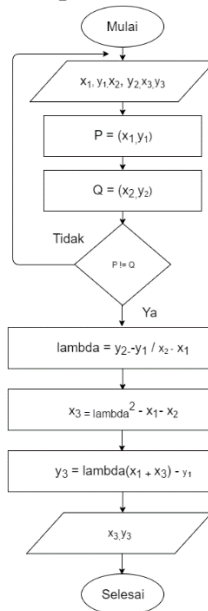
Ambill $P = (3,1)$ maka $2P = P + P = (x_3, y_3)$,perhitungannya seperti di bawah ini

$$\begin{aligned} x_3 &= \left[\left(\frac{3x_1^2 + a}{2y_1} \right)^2 - 2x_1 \right] \bmod p \\ &= \left[\left(\frac{3(3)^2 + 1}{2(1)} \right)^2 - 2(3) \right] \bmod 17 \\ &= \left[\left(\frac{28}{2} \right)^2 - 6 \right] \bmod 17 \\ &= [14^2 - 6] \bmod 17 \\ &= 190 \bmod 17 \\ &= 3 \end{aligned}$$

$$\begin{aligned} y_3 &= \left[\left(\frac{3x_1^2 + a}{2y_1} \right) (x_1 - x_3) - y_1 \right] \bmod p \\ &= \left[\left(\frac{3(3)^2 + 1}{2(1)} \right) (3 - 3) - 1 \right] \bmod 17 \\ &= \left[\left(\frac{28}{2} \right) (0) - 1 \right] \bmod 17 \\ &= [-1] \bmod 17 \\ &= 16 \end{aligned}$$

Sehingga didapat $(x_3, y_3) = (3, 16)$

Untuk mencari titik ketiga, tidak hanya menggunakan penjumlahan dua titik yang sama namun bisa menggunakan penjumlahan dua titik yang berbeda. Pada gambar 4.3 menjelaskan proses pembuatan titik ketiga di bidang terbatas f_p dengan penjumlahan dua titik yang berbeda. $P = (x_1, y_1)$ dan $Q = (x_2, y_2)$, untuk nilai $x_1, y_1, x_2, y_2, x_3, y_3 \in E(F_p)$. Syarat pertama untuk proses ini adalah $P \neq Q$, jika syarat tersebut tidak terpenuhi maka kembali ke awal untuk x_1, y_1, x_2, y_2 jika syarat terpenuhi maka ke proses selanjutnya yakni melakukan penghitungan menggunakan rumus lamda pada persamaan 2.6. Setelah nilai lamda ditemukan, kemudian x_3 dapat dihitung dengan menggunakan persamaan 2.10. Selanjutnya y_3 dapat dihitung dengan menggunakan persamaan 2.11. Titik ketiga x_3, y_3 telah ditemukan dari proses diatas



Gambar 4.3 Flowchart penambahan Titik jika $P \neq Q$

Contoh perhitungan secara aljabar untuk pembuatan titik ketiga dengan titik awal sama, untuk persamaan kurva eliptik dengan didapatkan titik kurva eliptik $y^2 = x^3 + x + 5$ dengan $a = 1, b = 5, p = 17$ didapatkan titik kurva eliptik

$$(x, y) = \{(2,7), (2,10), (3,1), (5,4), (5,13), (7,7), (7,10)\} \\ \{(8,7), (8,10), (11,2), (11,15), (14,3), (14,14)\}$$

Untuk $P = (x_1, y_1)$ dan $Q = (x_2, y_2)$ maka $P \neq Q = P + Q = R = (x_3, y_3)$ dimana :

$$x_3 = \left[\left(\frac{y_2 - y_1}{x_2 - x_1} \right)^2 - x_1 - x_2 \right] \bmod p \\ y_3 = \left[\left(\frac{y_2 - y_1}{x_2 - x_1} \right) (x_1 - x_3) - y_1 \right] \bmod p$$

Ambill $P = (3,1)$ dan $Q = (8,10)$ maka $P + Q = R = (x_3, y_3)$, perhitungannya seperti di bawah ini:

$$x_3 = \left[\left(\frac{y_2 - y_1}{x_2 - x_1} \right)^2 - x_1 - x_2 \right] \bmod p \\ = \left[\left(\frac{10 - 1}{8 - 3} \right)^2 - 3 - 8 \right] \bmod 17 \\ = \left[\left(\frac{10 - 1}{8 - 3} \right)^2 - 3 - 8 \right] \bmod 17 \\ = \left[\left(\frac{9}{5} \right)^2 - 11 \right] \bmod 17$$

Karena $9 \times 5^{-1} \bmod 17 = 9 \times 7 \bmod 17 = 12$ ini menghasilkan

$$= [12^2 - 11] \bmod 17 \\ = 14$$

$$y_3 = \left[\left(\frac{y_2 - y_1}{x_2 - x_1} \right) (x_1 - x_3) - y_1 \right] \bmod p$$

$$y_3 = \left[\left(\frac{10 - 1}{8 - 3} \right) (3 - 14) - 1 \right] \bmod 17$$

$$y_3 = \left[\left(\frac{9}{5} \right) (-11) - 1 \right] \bmod 17$$

$$y_3 = [12(-11) - 1] \bmod 17$$

$$y_3 = -133 \bmod 17 = 3$$

Sehingga didapat $(x_3, y_3) = (14, 3)$

4.2 Representasi Piksel pada Kurva Eliptik

Setiap citra terdiri dari piksel. Dalam citra *grayscale*, setiap piksel memiliki nilai 8-bit yang terdiri dari 0 sampai 255. Untuk mengenkripsi menggunakan sebuah citra dengan menggunakan ECC, setiap piksel dianggap adalah sebuah pesan dan harus dipetakan ke dalam setiap titik pada kurva eliptik yang sudah didefinisikan sebelumnya. Pada tugas akhir ini digunakan sebuah pemetaan berdasarkan sebuah tabel. Untuk membuat tabel ini, sebuah kurva eliptik $E_p(a, b)$ digenerasi semua titik yang ada, kemudian dipetakan menjadi 256 grup. Setiap grup mempunyai $N = \lceil \#E(f_p)/256 \rceil$ anggota. Setiap baris merupakan sebuah nilai piksel dan untuk nilai yang sama terdapat banyak titik. Jika N tidak mencapai 256 anggota, maka sisa kolom diisi dengan kosong sampai akhir[12].

Dari nilai piksel pertama dari sebuah citra plain, titik pada kurva eliptik yang berkorespondensi dengan piksel tersebut kemudian dipetakan hingga piksel terakhir. Misalkan pengirim dan penerima menentukan sebuah kurva eliptik $E_{7211}(1, 7206)$ yang direpresentasikan oleh persamaan 3.1:

$$y^2 \equiv x^3 + x + 7206 \mod 7211 \quad (4.1)$$

Tabel 4.3 menunjukkan titik titik yang digenerasi pada persamaan 3.1. Titik pertama disini adalah titik ideal yang akan berkorespondensi dengan nilai piksel 0, kemudian dilanjutkan dengan titik dan nilai piksel selanjutnya. Setelah 256 titik dimuat pada kolom pertama, 256 titik selanjutnya akan diletakkan pada kolom kedua, demikian selanjutnya sampai kolom yang terakhir. Dalam contoh ini terdapat 7223 titik pada kurva. Mengisi 29 kolom dan sisanya diisi dengan kosong.

Tabel 4.3 Tabel Pemetaan Piksel pada Titik Kurva Eliptik
 $E_{7211}(1,7206)$

Indeks	Pemetaan ke 1	Pemetaan ke 2	Pemetaan ke 3	--	Pemetaan ke 29
0	Ideal	(252,5914)	(553,6081)	--	(7162,4343)
1	(2,2467)	(253,1633)	(554,62)	--	(7163,336)
2	(2,4744)	(253,5578)	(554,7149)	-	(7163,6875)
3	(3,5)	(257,3158)	(555,1129)	--	(7166,1716)
4	(3,7206)	(257,4053)	(555,6083)	--	(7166, 5495)
5	(5,2087)	(259,1220)	(559,2336)	--	(7167,1536)
6	(5,5124)	(259,5991)	(559,4875)		(7167,5675)
7	(6,2616)	(260,287)	(560,1033)	--	(7168,2933)
--	--	--	--	--	--
154	(142,5136)	(421,6695)	(706,7042)	--	-
155	(145,3168)	(422,2950)	(707,1558)	--	-
156	(145,4043)	(422,4261)	(707,5653)	--	-
157	(146,592)	(423,27)	(709,1194)	--	-
158	(146,6619)	(423, 7184)	(709,6017)	--	-
159	(150,171)	(431, 2259)	(710,2454)	--	-
160	(150,7040)	(431,4682)	(710,4757)	--	-
161	(151,3116)	(432,2021)	(711,701)	--	-
--	--	--	--	--	--
248	(244,6910)	(541,4336)	(799,5288)	--	-
249	(245,56)	(542,867)	(802,3040)	--	-
250	(245,7155)	(542,6344)	(802,4171)	--	-
251	(248,195)	(548,2259)	(804,2396)	--	-
252	(248,7016)	(548,4952)	(804,4815)	--	-
253	(249,2367)	(549,1889)	(809,3157)	--	-
254	(249,4844)	(549,5322)	(809,4054)	--	-
255	(252,1297)	(553,1130)	(810,2214)	--	-

4.3 Pertukaran Kunci Diffie-Hellman

Dari parameter yang telah dibuat di subbab sebelumnya, diambil sebagian parameter untuk membangkitkan kunci publik dan kunci privat tersebut. Untuk membangkitkan kunci tersebut dapat menggunakan perhitungan $Q = d \cdot G$ yang sesuai dengan aturan kurva eliptik. Jika kunci sudah dibangkitkan, langkah selanjutnya yakni proses enkripsi dan dekripsi citra. Algoritma pembangkit kunci Diffie-Hellman dengan kurva eliptik yakni [11]

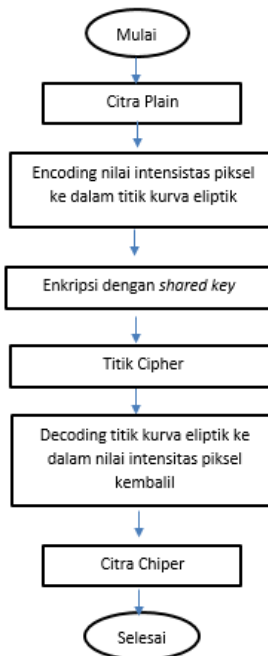
INPUT : Domain parameter $T(p, a, b, G, n, h)$
 OUTPUT : $K_{\text{publik}} = Q_1, Q_2$ $K_{\text{privat}} = d_1, d_2$
 Pilih $G = (x_1, y_1)$ sebagai titik pembangkit pada grup
 Kurva eliptik $E(a, b)$
 Pilih integer $d_1, d_2 \in [1, n - 1]$
 Hitung $Q_1 = d_1 \cdot G$ dan $Q_2 = d_2 \cdot G$
 $K_{\text{publik}} = Q_1, Q_2$ $K_{\text{privat}} = d_1, d_2$

4.4 Perancangan Sistem Enkripsi ECDH

Di dalam perancangan sistem ini, citra asli *grayscale* akan dienkripsi dengan menggunakan algoritma ECDH. Terlebih dulu untuk memulai proses enkripsi yakni dengan menentukan persamaan kurva eliptik (mod p), dengan memasukkan nilai dan untuk pemodulonya maka dari nilai tersebut dapat dihasilkan beberapa titik-titik yang sesuai dengan input yang dimasukkan. Nilai dan akan mempengaruhi .

Setelah menghasilkan beberapa titik tersebut, proses selanjutnya yakni memasukkan citra asli *grayscale* ke dalam sistem ini dengan menggunakan tabel pemetaan yang telah dibuat dari parameter-parameter kurva eliptik yang telah ditentukan. Proses selanjutnya yakni membangkitkan kunci (*generate key*).

Dimana kedua belah pihak membangkitkan kunci publik yang akan dimiliki keduanya dengan mengkalikan nilai G dengan kunci privat yang telah ditentukan oleh kedua belah pihak. Selanjutnya dihitung Kunci rahasia bersama yang didapat dari hasil perkalian kunci privat dengan pertukaran kunci publik kedua belah pihak. Untuk proses enkripsi, ambil setiap titik yang didapat dari piksel yang terpetakan ditambah dengan Kunci rahasia bersama untuk mencari titik-titik ketiga (x_3, y_3) , dimana titik ketiga ini merupakan chipper point pada kurva eliptik. Dengan menggunakan pemetaan tabel, kembali didapat nilai piksel dari titik chipper dan menjadi sebuah citra cipher.



Gambar 4.4 Diagram alir proses enkripsi pada citra

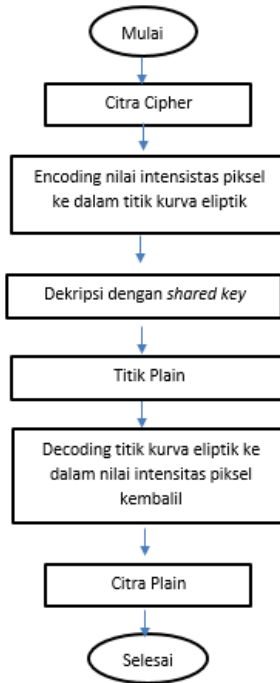
Citra asli (M) sebagai masukan algoritma enkripsi sistem kriptografi ECDH. Pengenkripsi memilih secara acak integer dan kemudian menghitungnya. Berikut adalah algoritma enkripsinya

4.5 Perancangan Sistem Dekripsi Citra ECDH

INPUT: Domain parameter $T(p, a, b, G, n, h)$, Kunci publik Q_1, Q_2 , Kunci Privat d_1, d_2 Titik Plain M
 OUTPUT : Titik Cipher C
 Hitung $SK = d_2 \cdot Q_1 = d_1 \cdot Q_2$
 Hitung $C = M + SK$
 Titik Cipher C

Pada perancangan sistem dekripsi ini, akan mengembalikan citra cipher menjadi sebuah citra plain kembali. Citra cipher akan dirubah kembali menjadi titik-titik yang ada di kurva eliptik dengan mengingat hasil titik cipher sebelumnya. Selanjutnya dilakukan proses dekripsi dengan mengurangi titik cipher dengan Kunci rahasia bersama. Setelah itu digunakan tabel pemetaan kembali untuk mengembalikan pada citra plain. Berikut adalah algoritma dekripsinya.

INPUT: Domain parameter $T(p, a, b, G, n, h)$, Kunci rahasia bersama SK Titik Cipher C
 OUTPUT : Titik Plain M
 Hitung $M = C - SK$
 Titik Plain M

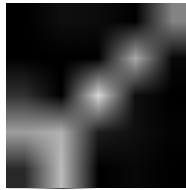


Gambar 4.5 Diagram alir proses dekripsi pada citra

4.6 Matriks yang dibentuk dari Citra dengan Format. JPG

File citra dengan format .jpg (Joint Photographics Group) termasuk format file citra raster yang sering dipakai. Citra raster adalah citra yang bergantung pada resolusi. Resolusi merupakan sebuah ekspresi $m \times n$ dimana m adalah jumlah baris dan n adalah jumlah kolom. Resolusi juga mengacu pada jumlah piksel di dalam sebuah citra. Pada Tugas akhir ini citra raster yang digunakan adalah sebuah citra grayscale.

Berikut akan diberikan contoh citra *grayscale* format .jpg dengan ukuran 5×5 yang dibentuk matriks dari citra tersebut.



Gambar 4.6. Citragray.jpg dengan ukuran 5×5

Dengan menggunakan Sage dan OpenCV (Sebuah library image processing pada Python), Citra tersebut dapat terbaca sebagai sebuah matriks $m \times n$ seperti sebagai berikut.

```
array([[ 0,  7,  6, 12, 132],
       [ 6,  0, 12, 174,  3],
       [50,  1, 198,  5,  1],
       [159, 177, 43,  5,  1],
       [25, 184,  3,  0,  4]], dtype=uint8)
```

Gambar 4.7. Matriks citragray.jpg pada SageMathCloud dan OpenCV

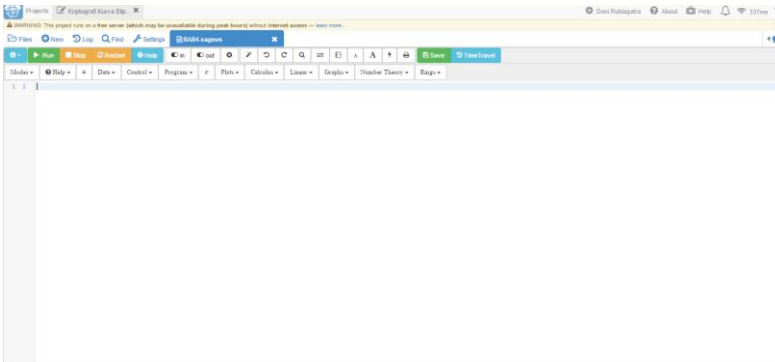
4.7 Implementasi pada SageMathCloud

4.7.1 SageMathCloud (SMC)

SageMathCloud adalah sebuah software matematika berbasis web yang *open-source* dan memiliki lisensi GPL. SageMathCloud hampir dibuat dengan 100 paket *open-source*. SageMathCloud dapat digunakan dalam pembelajaran matematika dasar hingga mahir. Hal ini mencakup Aljabar, Kalkulus, Teori Bilangan, Kriptografi, Komputasi Numerik, Aljabar Komutatif, Teori Grup, Kombinatorik, Teori Graf, Aljabar Linier, dan masih banyak yang lain. Dengan banyaknya

macam paket software dan integrasi fungsinya, SageMathCloud sangat cocok untuk keperluan edukasi dan penelitian.

Interface pada SageMathCloud merupakan web browser maupun sebuah *command line*. Dengan menggunakan *notebook* yang tersedia kita dapat membuat sebuah grafis maupun penulisan ekspresi matematika yang cantik.



Gambar 4.8 *Interface SageMathCloud notebook*

Setelah kita *log in* ke dalam *SageMathCloud*, Pertama dibuat class-class seperti pada gambar 4.9 seperti berikut.



Gambar 4.9 Tampilan nama program python yang dibuat

4.7.2 Pembacaan Piksel dalam Citra

Pertama, kita *import* beberapa *class* dan *library* yang dibutuhkan. Pada tahap ini citra asli *grayscale* gambar 4.10 sebagai file input akan dibaca oleh SMC yang hasilnya berupa sebuah matriks berukuran $m \times n$. Kemudian matriks itu akan diubah menjadi sebuah list agar dapat mudah untuk diolah dapat dilihat pada gambar 4.11



Gambar 4.10 Lena100.jpg citra *grayscale* berukuran 100×100

```

2 1
3 2      ### Import Library ###
4 3      import cv2
5 4      import numpy as np
6 5      import math
7 6      import random
8 7      import time
9 8      from kurvaeliptik import *
10 9      from semuatitik import *
11 10
12 11      ### Load Lena Images and convert it to list ###
13 12      lena = cv2.imread('Lenna100.jpg')
14 13      lenagray = cv2.cvtColor(lena, cv2.COLOR_BGR2GRAY)
15 14      lenagray
16 15      x = lenagray.tolist()
17
      array([[171, 168, 166, ..., 113, 145, 169],
             [166, 164, 163, ..., 135, 93, 56],
             [164, 164, 164, ..., 89, 39, 11],
             ...,
             [ 37, 44, 75, ..., 40, 30, 35],
             [ 26, 36, 51, ..., 26, 35, 63],
             [ 31, 29, 46, ..., 41, 72, 89]], dtype=uint8)

```

Gambar 4.11 *Import library* dan pembacaan piksel pada citra

4.7.3 Pembuatan Titik Kurva Eliptik beserta Tabel Pemetaan

Pada proses enkripsi dan dekripsi citra dengan menggunakan algoritma ECDH yang mana untuk dibutuhkan sebuah tabel pemetaan untuk merubah piksel menjadi titik di kurva eliptik, maka diperlukan terlebih dahulu untuk mencari semua titik yang digunakan pada kurva eliptik tersebut. Pada program enkripsi dan dekripsi citra ini menggunakan metode kriptografi kurva eliptik pada bidang terbatas F_p yang perhitungannya sesuai dengan persamaan 2.5 yang mana $a, b \in F_p$, dan p adalah bilangan prima ganjil dengan $p > 3$. Untuk perhitungan tersebut jika diterapkan dalam pemrograman Python menjadi seperti pada gambar 4.12, 4.13 dan 4.14 berikut.

```
class KurvaEliptik(object):
    def __init__(self, a, b):
        # assume we're already in the Weierstrass form
        self.a = a
        self.b = b

        self.diskriminan = -16 * (4 * a*a*a + 27 * b * b)
        if not self.cekDiskriminan():
            raise Exception("kurva Eliptik %s Memiliki diskriminan = 0" % self)

    def cekDiskriminan(self):
        return self.diskriminan != 0

    def ujiTitik(self, x, y):
        return y*y == x*x*x + self.a * x + self.b

    def __str__(self):
        return 'Kurva Eliptik y^2 = x^3 + %sx + %s' % (self.a, self.b)

    def __repr__(self):
        return str(self)

    def __eq__(self, other):
        return (self.a, self.b) == (other.a, other.b)
```

Gambar 4.12 *Class KurvaEliptik*

Untuk lapangan terbatas yang digunakan adalah F_{7211} dengan nilai $a = 1$, dan $b = 7206$, sehingga kurva eliptik terdefinisi sebagai $E_{7211}(1,7206)$. Kemudian dipilih sebuah titik generator $G = (3,5)$.

```
class Titik(object):
    def __init__(self, kurva, x, y):
        self.kurva = kurva # the kurva containing this Titik
        self.x = x
        self.y = y

        if not kurva.ujiTitik(x,y):
            raise Exception("Titik %s tidak terdapat dalam kurva %s!" % (self, kurva))

    def __str__(self):
        return("(%r, %r)" % (self.x, self.y))

    def __repr__(self):
        return str(self)

    def __neg__(self):
        return Titik(self.kurva, self.x, -self.y)
```

Gambar 4.13 Class Titik pada kurva eliptik

```
from kurvaeliptik import *
def semuaTitik(kurva, a, b, besar, field):
    xs = []
    ys = []
    points = []
    points.append(Ideal(kurva))
    for i in range(besar):
        for j in range(besar):
            hasilb = (i**3 + a * i + b) % besar
            hasila = (j**2) % besar
            if hasila == hasilb:
                xs.append((field(i)))
                ys.append((field(j)))

    for x, y in zip(xs,ys):
        points.append((Titik(kurva,x,y)))

    return points
```

Gambar 4.14 Fungsi semuaTitik

```

### Elliptic Curve, base and all point Initialization###
F7211 = IntegersModP(7211)
E = KurvaEliptik(F7211(1), F7211(7206))
G = Titik(E, F7211(3), F7211(5))
all_points = semuaTitik(E, 1, 7206, 7211, F7211)
E
G

Kurva Eliptik  $y^2 = x^3 + 1x + 7206$ 
(3 (mod 7211), 5 (mod 7211))

```

Gambar 4.15 Pendefinisian kurva eliptik pada SMC

Setelah mendefinisikan Kurva eliptik beserta semua titik seperti pada gambar 4.15, kemudian kita membuat sebuah tabel pemetaan dengan ukuran yang sama dengan jumlah dari semua titik kurva eliptik yang didefinisikan oleh E_{7211} (1,7206) proses tersebut dapat dilihat pada gambar 4.16.

```

### Makes a Map from image to point in Elliptic Curve ###

pixelto = []
pixelto_gen = dict.fromkeys(range(0, 256))
pixelto_gen1 = dict.fromkeys(range(0,55))

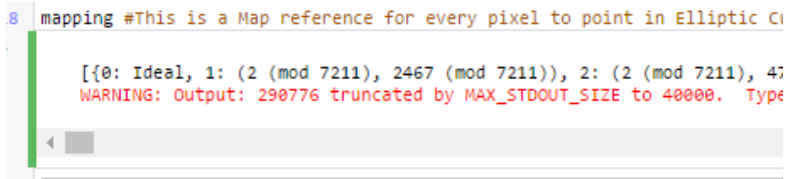
for i in range(28):
    pixelto.append(pixelto_gen)
    pixelto.append(pixelto_gen1)

mapping = []
for i, isi in enumerate(pixelto):
    mapping.append({})
    for j, konten in enumerate(isi):
        mapping[i][konten] = all_points[(i*256)+int(konten)]

```

Gambar 4.16 Pembuatan Tabel Pemetaan pada SMC

Dengan jumlah titik pada kurva eliptik E_{7211} (1,7206) mencapai 7223, maka terdapat 28 grup pemetaan dengan piksel lengkap yaitu 256 buah ditambah 1 grup pemetaan dengan 55 buah piksel. Dapat dilihat hasilnya pada gambar 4.17



```
8 mapping #This is a Map reference for every pixel to point in Elliptic C
[0: Ideal, 1: (2 (mod 7211), 2467 (mod 7211)), 2: (2 (mod 7211), 47
WARNING: Output: 290776 truncated by MAX_STDOUT_SIZE to 40000. Type
```

Gambar 4.17 Potongan hasil Tabel Pemetaan pada SMC

4.7.4 Pertukaran Kunci Diffie-Hellman

Untuk pertukaran kunci Diffie-Hellman, kita misalkan terdapat dua orang A dan B . Mereka berdua sama-sama mengetahui parameter-parameter yang digunakan dalam algoritma ECDH. Kemudian mereka memilih sebuah kunci privat masing-masing didefinisikan dengan N_A dan N_B . Untuk mendapatkan kunci publik mereka mengalikan kunci privat mereka dengan titik G , kemudian saling bertukar. Setelah bertukar kunci publik, mereka mengalikan kembali kunci publik tersebut dengan kunci privatnya masing-masing.

Pertukaran kunci seperti ini membuat dua orang A dan B dapat berkomunikasi walaupun di dalam jaringan yang tidak aman. Untuk mengaplikasikan sebuah pertukaran kunci Diffie-Hellman, kita harus menerapkan sebuah multiplikasi skalar pada kurva eliptik. Jika P adalah sebuah titik pada kurva eliptik, dan n adalah sebuah integer maka $nP = P + P + P + \dots$ hingga n kali. Algoritma Diffie-Hellman dapat dilihat pada gambar 4.18

```

def __mul__(self, n):
    if not isinstance(n, int):
        raise Exception("Tidak bisa selain integer untuk multiplikasi skalar")
    else:
        if n < 0:
            return -self * -n
        if n == 0:
            return Ideal(self.kurva)
        else:
            Q = self
            R = self if n & 1 == 1 else Ideal(self.kurva)

            i = 2
            while i <= n:
                Q = Q + Q

                if n & i == i:
                    R = Q + R

                i = i << 1

            return R

```

Gambar 4.18 Listing program multiplikasi skalar untuk Titik

```

### Diffie Hellman Key Exchange
N_A = 12
N_B = 23
A = int(N_A) * G
B = int(N_B) * G
A
B

(1794 (mod 7211), 6375 (mod 7211))
(3861 (mod 7211), 1242 (mod 7211))

K_A = B * int(N_A)
K_B = A * int(N_B)
K_A
K_B

(1472 (mod 7211), 2098 (mod 7211))
(1472 (mod 7211), 2098 (mod 7211))

```

Gambar 4.19 Implementasi Diffie-Hellman pada SMC

Dapat dilihat pada gambar 4.19 mereka mendapatkan Kunci rahasia bersama yang mempunyai nilai yang sama. Kunci inilah yang akan digunakan dalam proses enkripsi maupun dekripsi.

4.7.5 Proses Pemetaan dan Enkripsi Citra

Setelah mendefinisikan kurva eliptik dan membuat tabel pemetaan, maka selanjutnya adalah proses enkripsi pada citra. Sebelum dilakukan proses enkripsi, setiap piksel akan dianggap sebagai pesan untuk dienkripsi. Sehingga, setiap piksel yang ada akan direpresentasikan pada titik di kurva eliptik. Berikut adalah implementasinya pada gambar 4.20.

```

### Mapping process from image to Elliptic Curve ###
rnd = random.choice(mapping)
x_mapping = [[]]
for i, isi in enumerate(x):
    if i>0:
        x_mapping.append([])
        for j, konten in enumerate(isi):
            if konten in rnd:
                x_mapping[i].append(rnd[konten])
x_mapping

[[[969 (mod 7211), 918 (mod 7211)), (966 (mod 7211), 5729 (mod 7211)]]
WARNING: Output: 349352 truncated by MAX_STDOUT_SIZE to 40000.

```

Gambar 4.20 Proses pemetaan setiap piksel pada SMC

Setelah dilakukan pemetaan, maka setiap piksel akan menjadi sebuah titik yang berada pada kurva eliptik. Kemudian setiap titik yang ada akan dienkripsi menggunakan kunci rahasia bersama yang telah dibuat sebelumnya. Dengan menambahkan titik awal piksel dengan titik pada kunci rahasia bersama. Program fungsi. Untuk proses penambahan kurva eliptik, algoritma dapat dilihat pada gambar 4.21

```

def __add__(self, Q):
    if self.kurva != Q.kurva:
        raise Exception("Tidak bisa menambahkan titik dengan kurva yang berbeda")
    if isinstance(Q, Ideal):
        return self

    x_1, y_1, x_2, y_2 = self.x, self.y, Q.x, Q.y

    if (x_1, y_1) == (x_2, y_2):
        if y_1 == 0:
            return Ideal(self.kurva)

        m = (3 * x_1 * x_1 + self.kurva.a) / (2 * y_1)
    else:
        if x_1 == x_2:
            return Ideal(self.kurva)

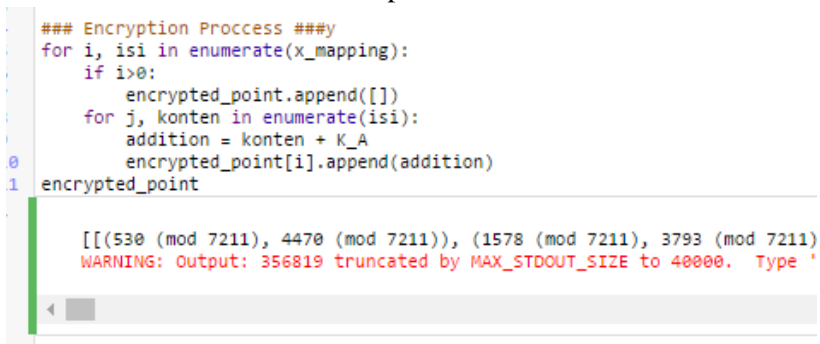
        m = (y_2 - y_1) / (x_2 - x_1)

    x_3 = m*m - x_2 - x_1
    y_3 = m*(x_3 - x_1) + y_1

    return Titik(self.kurva, x_3, -y_3)

```

Gambar 4.21 Listing program penambahan dua titik pada kurva eliptik



```

### Encryption Process ###
for i, isi in enumerate(x_mapping):
    if i>0:
        encrypted_point.append([])
        for j, konten in enumerate(isi):
            addition = konten + K_A
            encrypted_point[i].append(addition)
0 encrypted_point
1

[[ (530 (mod 7211), 4470 (mod 7211)), (1578 (mod 7211), 3793 (mod 7211)) ]
WARNING: Output: 356819 truncated by MAX_STDOUT_SIZE to 40000. Type '

```

Gambar 4.22 Proses enkripsi pada SMC

Pada gambar 4.22 dapat kita lihat bahwa titik awal (titik plain) ditambahkan dengan kunci bersama menghasilkan sebuah titik cipher atau titik yang telah terenkripsi. Kemudian untuk mengembalikan titik menjadi sebuah piksel prosesnya dapat dilihat pada gambar 4.23 dan hasil citra yang telah terenkripsi dapat dilihat pada gambar 4.24

```

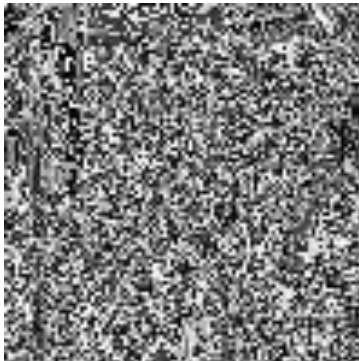
    ### Encrypted Image ###
    start = time.time()
    encrypted_image = [[]]
    for i, isi in enumerate(encrypted_point):
        if i>0:
            encrypted_image.append([])
            for j, konten in enumerate(isi):
                for k in mapping:
                    for l in k:
                        if konten == k[l]:
                            encrypted_image[i].append(l)
    end = time.time()
    print end - start

    381.303055048

    encrypted_image
    cv2.imwrite('lena100TESSELF_encrypted.jpg',np.array(encrypted_image))

```

Gambar 4.23 Proses pengembalian titik cipher menjadi citra kembali



Gambar 4.24 Hasil citra lenna100 yang telah terinkripsi

4.7.5 Proses Pemetaan dan Dekripsi Citra

Pada bagian ini, citra digital yang telah menjadi citra terenkrip kemudian dilakukan proses dekripsi atau pengembalian ke dalam bentuk semula (citra asli). Berikut adalah listing programnya.

```
### Decryption Proccess ###

decrypted_point = [[]]
for i, isi in enumerate(encrypted_point):
    if i>0:
        decrypted_point.append([])
        for j, konten in enumerate(isi):
            subtraction = konten - K_B
            decrypted_point[i].append(subtraction)
```

```
decrypted_point
```

```
[[ (969 (mod 7211), 918 (mod 7211)), (966 (mod 7211), 5729 (mod 7211)
WARNING: Output: 349352 truncated by MAX_STDOUT_SIZE to 40000. Typ
```

Gambar 4.25 Proses dekripsi menjadi titik plain kembali pada SMC

Dapat dilihat pada gambar 4.25, seperti halnya sebuah enkripsi, kunci rahasia bersama digunakan untuk mendekripsi citra tersebut.. Proses dekripsi ini adalah dengan mengambil setiap piksel yang ada di citra cipher kemudian mengurangnya dengan kunci rahasia bersama, sehingga didapatkan sebuah titik yang sama seperti pada awal citra plain. Kunci rahasia kali ini yang digunakan adalah yang dimiliki oleh user *B*. Kemudian pada gambar 4.26 merupakan proses pengembalian titik menjadi piksel kembali dan gambar 4.27 merupakan hasil citra dekripsi.

```

start = time.time()
decrypted_image = [[]]
for i, isi in enumerate(decrypted_point):
    if i>0:
        decrypted_image.append([])
        for j, konten in enumerate(isi):
            for k in mapping:
                for l in k:
                    if konten == k[l]:
                        decrypted_image[i].append(l)
end = time.time()
print end - start

327.108179092
327.108179092

cv2.imwrite('lena100TESSELF_decrypted.jpg',np.array(decrypted_image))

True

```

Gambar 4.26 Proses pengembalian titik dekripsi menjadi citra kembali



Gambar 4.27 Citra hasil dekripsi

“Halaman ini sengaja dikosongkan”

BAB V

PENGUJIAN DAN PEMBAHASAN

Pada bab ini berisi tentang hasil uji coba yang dihasilkan oleh implementasi sistem dan melakukan pembahasan terhadap hasil uji coba yang dilakukan.

5.1 Pengujian Titik Kurva Eliptik

Setelah dilakukan pembuatan titik kurva eliptik pada program, maka selanjutnya yang dilakukan adalah pengujian terhadap titik dengan data yang akan diambil berbeda – beda untuk nilai a, b dan p nya. Pada tabel 5.1 akan ditampilkan hasil pengujian titik kurva eliptik dengan nilai a, b dan p dan yang berbeda.

Tabel 5.1 Pengujian titik kurva eliptik

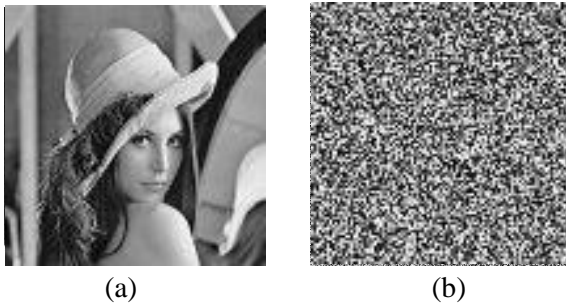
No	Nilai a, b dan p	Hasil Titik dalam Program	No	Nilai a, b dan p	Hasil Titik dalam Program
1	$a = 13$ $b = 5$ $p = 11$	$x = 0, y = 4$ $x = 0, y = 7$ $x = 3, y = 4$ $x = 3, y = 7$ $x = 4, y = 0$ $x = 8, y = 4$ $x = 8, y = 7$ $x = 9, y = 2$ $x = 4, y = 9$	2.	$a = 3$ $b = 4$ $p = 11$	$x = 0, y = 2$ $x = 0, y = 9$ $x = 4, y = 5$ $x = 4, y = 6$ $x = 5, y = 1$ $x = 5, y = 10$ $x = 7, y = 4$ $x = 7, y = 7$ $x = 8, y = 1$ $x = 8, y = 10$ $x = 9, y = 1$ $x = 9, y = 10$ $x = 10, y = 0$

3	$a = 5$ $b = 6$ $p = 13$	$x = 1, y = 5$ $x = 1, y = 8$ $x = 3, y = 3$ $x = 3, y = 10$ $x = 4, y = 5$ $x = 4, y = 8$ $x = 5, y = 0$ $x = 8, y = 5$ $x = 8, y = 8$ $x = 9, y = 0$ $x = 10, y = 4$ $x = 10, y = 9$ $x = 11, y = 1$ $x = 11, y = 12$ $x = 12, y = 0$	4	$a = 7$ $b = 20$ $p = 23$	$x = 6, y = 5$ $x = 6, y = 18$ $x = 8, y = 6$ $x = 8, y = 17$ $x = 10, y = 3$ $x = 10, y = 20$ $x = 11, y = 5$ $x = 13, y = 18$ $x = 13, y = 10$ $x = 15, y = 13$ $x = 15, y = 2$ $x = 15, y = 21$ $x = 20, y = 8$ $x = 20, y = 15$ $x = 22, y = 9$ $x = 22, y = 14$
5	$a = 1$ $b = 1$ $p = 23$	$x = 0, y = 1$ $x = 0, y = 22$ $x = 1, y = 7$ $x = 1, y = 16$ $x = 3, y = 10$ $x = 3, y = 13$ $x = 4, y = 0$ $x = 5, y = 4$ $x = 5, y = 19$ $x = 6, y = 4$ $x = 6, y = 19$ $x = 7, y = 11$ $x = 7, y = 12$ $x = 9, y = 7$ $x = 9, y = 16$ $x = 11, y = 3$ $x = 11, y = 20$ $x = 12, y = 4$ $x = 12, y = 19$ $x = 13, y = 7$ $x = 13, y = 16$ $x = 17, y = 3$ $x = 17, y = 20$ $x = 18, y = 3$ $x = 18, y = 20$ $x = 19, y = 5$ $x = 19, y = 18$	6	$a = 7$ $b = 20$ $p = 31$	$x = 0, y = 12$ $x = 0, y = 19$ $x = 1, y = 11$ $x = 1, y = 20$ $x = 4, y = 9$ $x = 4, y = 22$ $x = 5, y = 5$ $x = 5, y = 26$ $x = 7, y = 3$ $x = 7, y = 28$ $x = 10, y = 6$ $x = 10, y = 25$ $x = 11, y = 8$ $x = 11, y = 23$ $x = 13, y = 13$ $x = 13, y = 18$ $x = 14, y = 14$ $x = 14, y = 17$ $x = 15, y = 11$ $x = 15, y = 20$ $x = 20, y = 10$ $x = 20, y = 21$ $x = 21, y = 2$ $x = 21, y = 29$ $x = 23, y = 14$ $x = 23, y = 17$ $x = 24, y = 0$ $x = 25, y = 14$ $x = 25, y = 17$

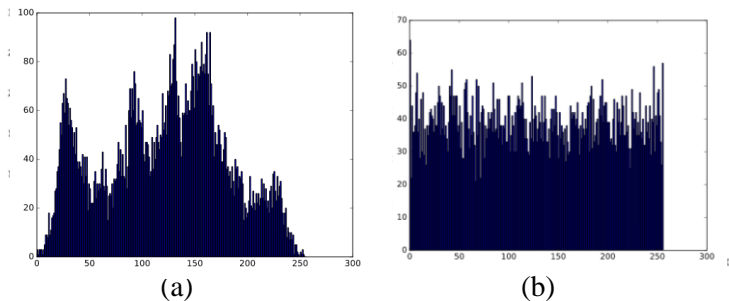
5.2 Pengujian Citra Hasil Enkripsi

Untuk menentukan apakah sebuah kriptosistem kuat atau tidak, maka dilakukan sebuah beberapa bentuk analisis dan uji coba. Terdapat beberapa analisis dengan berbagai cara dari serangan kriptografi.

Analisis Histogram: Sebuah histogram adalah sebuah distribusi grafis dari intensitas setiap nilai piksel. Untuk mencegah bocornya informasi dari citra cipher, sebuah histogram citra cipher harus berbeda secara signifikan dengan citra plainnya. Pengujian menggunakan citra Lena ukuran 100×100 dan menggunakan persamaan 4.1



Gambar 5.1 (a) Citra plain, (b) Citra cipher



Gambar 5.2 (a) Histogram citra plain,
(b) Histogram citra cipher

Analisis Korelasi: Dua piksel yang berdekatan dalam citra plain memiliki korelasi yang kuat secara vertikal, horizontal maupun diagonal. Nilai korelasi ini dalam citra biasa, memiliki

nilai maksimum 1 dan nilai minimum -1. Sebuah citra yang terenkripsi dengan baik agar menghindari serangan secara statistik, harus memiliki nilai korelasi yang mendekati 0. Kovarians dari x dan y dihitung dengan menggunakan persamaan 5.1 dan varians x dan y dihitung dengan persamaan 5.2. Terakhir, menggunakan kovarians dan standar deviasi untuk menghitung korelasi koefisien $r_{x,y}$ dengan persamaan 5.3. Untuk hasil perbandingan antara citra plain dan citra cipher dapat dilihat pada tabel 3.1. Distribusi korelasi untuk vertikal, horizontal dan diagonal dari piksel yang berdekatan

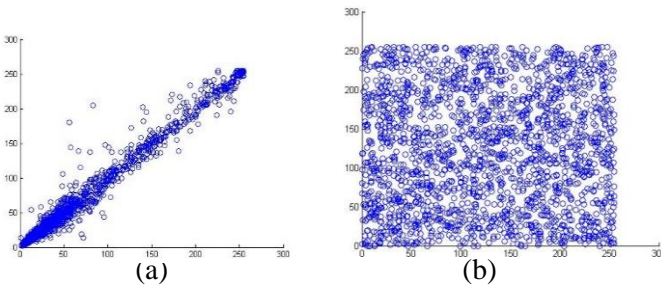
$$D(x) = \frac{1}{N} \sum_{i=1}^N ((x_i - E(x))^2), E(x) = \frac{1}{N} \sum_{i=1}^N x_i \quad (5.1)$$

$$cov(x, y) = \frac{1}{N} \sum_{i=1}^N (x_i - E(x))(y_i - E(y)) \quad (5.2)$$

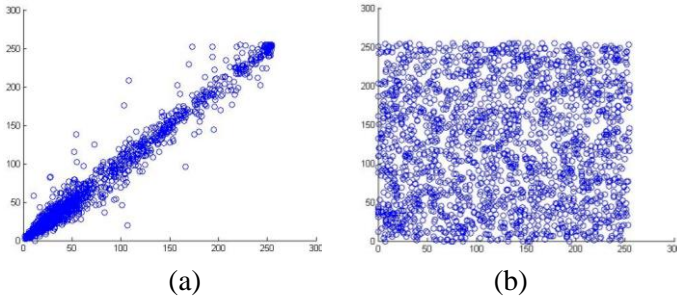
$$r_{x,y} = \frac{cov(x)}{\sqrt{D(x)}\sqrt{D(y)}} \quad (5.3)$$

Tabel 5.2 Nilai korelasi citra plain dan citra cipher

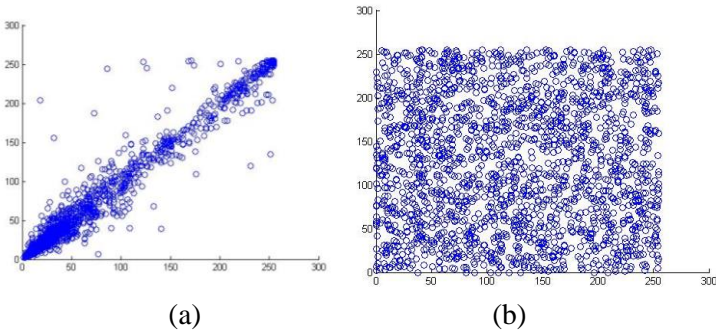
	Korelasi Horizontal	Korelasi Vertikal	Korelasi Diagonal
Citra Plain	0.9910	0.9906	0.9797
Citra Cipher	-0.0165	0.0180	0.0182



Gambar 5.3 Scatter plot korelasi horizontal, (a) Citra plain, (b) Citra cipher



Gambar 5.4 Scatter Plot Korelasi Vertikal, (a) Citra Plain, (b) Citra Citpher

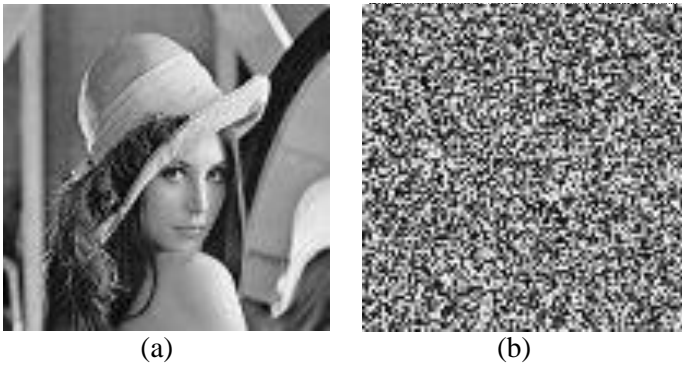


Gambar 5.5 Scatter Plot Korelasi Diagonal, (a) Citra Plain, (b) Citra Citpher

Analisis Entropi: Entropi adalah sebuah parameter statistik skalar yang mengukur tingkat seberapa acak sebuah citra. Nilai tertinggi adalah 8, hal ini berarti sebuah Citra Cipher memiliki tekstur yang acak. Entropi dari sebuah citra terenkripsi yang diusulkan melalui skema ini dihitung menggunakan persamaan 5.4 ini memiliki hasil 7.9894

$$\text{Entropy} = \sum_{i=0}^n P_i \log_2 P_i \quad (5.4)$$


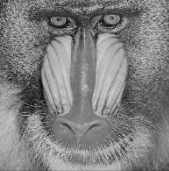

Analisis sensitifitas kunci Sebuah Algoritma yang aman memiliki tingkat sensitifitas tinggi terhadap kunci. Ini berarti bahwa citra yang telah terenkripsi tidak dapat didekripsi walaupun kuncinya dirubah dengan nilai yang kecil. Pada gambar 5.5 mengganti nilai kunci untuk dekripsi dan menggunakan kunci ini untuk mendekripsi akan menghasilkan perubahan yang ekstrim. Hal ini membuat sebuah citra enkripsi tahan terhadap serangan *brute-force attack*.



Gambar 5.6 (a) Citra Dekripsi dengan kunci sesuai $k = 2345$,
(b) Citra Dekripsi dengan kunci $k = 2346$

5.4 Pengujian Berbagai Ukuran dan Macam Citra

Tabel 5.3 Hasil uji coba berbagai jenis citra dan ukuran

Nama Citra	Percobaan Ke -	Ukuran Citra	Waktu Enkripsi	Waktu Dekripsi
Lena.jpg 	1	50 × 50	87	23
	2	100 × 100	158	94
	3	200 × 200	442	379
	4	300 × 300	916	854
	5	400 × 400	1581	1521
		Rata-Rata	1919,2	1654,2
Mandrill.jpg 	1	50 × 50	89	23
	2	100 × 100	160	95
	3	200 × 200	446	382
	4	300 × 300	924	860
	5	400 × 400	1590	1530
		Rata-Rata	1937	1666
Cameraman.jpg 	1	50 × 50	88	23
	2	100 × 100	159	95
	3	200 × 200	444	383
	4	300 × 300	920	855
	5	400 × 400	1596	1535
		Rata-Rata	1930,2	1663

Dapat dilihat pada tabel 5.3, uji coba dilakukan dengan total 15 kali percobaan dengan ukuran dan macam citra yang berbeda, serta parameter waktu dalam *second*. Untuk proses enkripsi dengan ukuran citra yang kecil menghasilkan waktu enkripsi yang memiliki rata-rata kecil juga. Sedangkan untuk proses enkripsi dengan ukuran citra yang cukup besar mempunyai rata-rata waktu yang besar yang artinya proses tersebut membutuhkan waktu yang cukup lama. Sama halnya dengan proses dekripsi dengan ukuran dan macam citra yang berbeda.

“Halaman Sengaja Dikosongkan”

BAB VI

KESIMPULAN DAN SARAN

Bab ini berisi tentang beberapa kesimpulan yang dihasilkan berdasarkan penelitian yang telah dilaksanakan. Di samping itu, pada bab ini juga dimasukkan beberapa saran yang dapat digunakan jika penelitian ini ingin dikembangkan.

6.1 Kesimpulan

Berdasarkan hasil perancangan sistem dan uji coba program, dapat diambil beberapa kesimpulan sebagai berikut:

1. Dari hasil pengujian titik kurva eliptik terlihat titik-titik yang dihasilkan dipengaruhi oleh nilai a, b dan p nya. Sehingga semakin besar nilai p nya maka titik yang dihasilkan pun semakin banyak. Namun dengan p yang sama, tetapi nilai a, b semakin besar maka jumlah titik akan semakin kecil. Hal ini dikarenakan nilai a, b dan p merupakan nilai yang digunakan untuk membangkitkan titik-titik kurva tersebut.
2. Semakin besar lapangan terbatas yang digunakan maka elemen elemen disana pun semakin panjang, sehingga membuat tabel pemetaan akan semakin besear. Hal ini membuat proses pengembalian citra dari titik kurvva eliptik akan memiliki waktu komputasi yang semakin lama.
3. Berdasrkan hasil pengujian, cintra enkripsi sangat kuat terhadap beberapa serangan kriptografi. Hal ini dapat diketahui pada hasil analisis histogram, korelasi, entropi dan sensitifitas kunci dari citra enkripsi .

6.2 Saran

Berdasarkan hasil yang sudah dicapai pada tugas akhir ini, terdapat beberapa hal yang perlu dipertimbangkan untuk pengembangannya antara lain:

1. Program enkripsi-dekripsi ini masih bersifat statis, sehingga untuk melakukan uji coba dengan variabel berbeda harus diganti secara manual maka untuk penelitian selanjutnya diharapkan bersifat dinamis agar program lebih mempermudah user dalam melakukan proses enkripsi-dekripsi citra.
2. Mengembangkan dengan file input citra berbagai ukuran dan format.
3. Karena pada skema kriptosistem ini mengharuskan mengkomputasi tiap pikselnya untuk dienkripsi, maka kedepannya diharapkan ada proses grouping beberapa piksel menjadi satu untuk membuat proses komputasi menjadi lebih efisien.
4. Tidak dipungkiri dalam proses transmisi mungkin terdapat noise yang tersemat pada citra dan membuat nilai piksel pada citra enkripsi akan berubah. Hal ini berakibat citra tidak akan bisa didekripsi karena sistem membutuhkan nilai piksel yang sama ketika enkripsi telah berhasil. Untuk penelitian selanjutnya disarankan dapat membuat sebuah sistem tersendiri untuk mengatasi adanya *noise* seperti ini.

DAFTAR PUSTAKA

- [1] Guo, C., et al. (2014). "Optical Double Image Encryption Employing A Pseudo Image Technique In The Fourier Domain". **Opt.Commun**, Vol. **321**, 61–72.
- [2] Yang, Y.-G., et al. (2014). "Quantum Cryptographic Algorithm For Color Images Using Quantum Fourier Transform and Double Random-Phase Encoding". **Information Sciences**, Vol. **277**, 445–457.
- [3] Bouguezal, S. (2012). "A Reciprocal-Orthogonal Parametric Transform And Its Fast Algorithm". **IEEE Signal Process, Lett.** **19**(11), 769–772
- [4] Zhou, Y., et al. (2012). "Image Encryption using P-Fibonacci Transform and Decomposition". **Optics Communications** Vol. **285**, 594–608.
- [5] Liao, X., et al. (2010). "A Novel Image Encryption Algorithm based Onself-adaptive Wave Transmission". **Signal Process**, Vol. **90**, 2714–2722..
- [6] Singh, L.D & Singh, K.M. (2015). "Image Encryption pusing Elliptic Curve Cryptography". **Procedia Computer Science**, Vol **54**, 472-481.
- [7] Liu, Hong & Liu, Yanbing. (2014). "Cryptanalyzing an Image Encryption Scheme based on Hybrid Chaotic System Ana Cyclic Elliptic Curve". **Optics & Laser Technology**, Vol. **56**, 15-19

- [8] Bao, Long & Zhou Yicong. (2015). "Image Encryption: Generating Visually Meaningful Image". **Information Sciences, Vol. 324, 197-207.**
- [9] Gonzalez, R.C., et al. (2009). "Digital Image Processing using Matlab". **Gatesmark. USA.**
- [10] Stallings, William. (2014). "Cryptography And Network Security Principles and Practice Sixth Edition". **Pearson. USA**
- [11] Cohen, Henri & Frey, Gerhard. (2006). "Handbook of Elliptic And Hyperelliptic Curve Cryptography". **Chapman & Hall/CRC. USA**
- [12] Solyeami, Ali., et al. (2013). "A Novel Public Key Image Encryption Based on Elliptic Curves over Prime Group Field". **Journal of Image and Graphics,, Vol.1, 43–49.**

LAMPIRAN

1. Potongan kode KurvaEliptik.py

```
from modp import *

class KurvaEliptik(object):
    def __init__(self, a, b):
        self.a = a
        self.b = b

        self.diskriminan = 16 * (4 * a*a*a
                                + 27 * b * b)
        if not self.cekDiskriminan():
            raise Exception("Kurva Eliptik %s Memiliki
                            diskriminan = 0" % self)

    def cekDiskriminan(self):
        return self.diskriminan != 0

    def ujiTitik(self, x, y):
        return y*y == x*x*x + self.a * x + self.b

    def __str__(self):
        return 'Kurva Eliptik  $y^2 = x^3 + ax + b$ ' % (self.a, self.b)

    def __repr__(self):
        return str(self)

    def __eq__(self, other):
        return (self.a, self.b) == (other.a, other.b)
```

```

class Titik(object):
    def __init__(self, kurva, x, y):
        self.kurva = kurva
        self.x = x
        self.y = y

    if not kurva.ujiTitik(x,y):
        raise Exception("Titik %s tidak terdapat
                        dalam kurva %s!"
                        % (self, kurva))

    def __str__(self):
        return("(%r, %r)" % (self.x, self.y))

    def __repr__(self):
        return str(self)

    def __neg__(self):
        return Titik(self.kurva, self.x, -self.y)

    def __add__(self, Q):
        if self.kurva != Q.kurva:
            raise Exception("Tidak bisa menambahkan titik
                            dengan kurva yang berbeda")
        if isinstance(Q, Ideal):
            return self

        x_1, y_1, x_2, y_2 = self.x, self.y, Q.x, Q.y

        if (x_1, y_1) == (x_2, y_2):
            if y_1 == 0:
                return Ideal(self.kurva)

            m = (3 * x_1 * x_1 + self.kurva.a)
                / (2 * y_1)
        else:

```

```

if x_1 == x_2:
    return Ideal(self.kurva)

    m = (y_2 - y_1) / (x_2 - x_1)

    x_3 = m*m - x_2 - x_1
    y_3 = m*(x_3 - x_1) + y_1

    return Titik(self.kurva, x_3, -y_3)

def __sub__(self, Q):
    return self + -Q

def __mul__(self, n):
if not isinstance(n, int):
    raise Exception("Tidak bisa selain integer
                    untuk multiplikasi skalar")
else:
    if n < 0:
        return -self * -n
    if n == 0:
        return Ideal(self.kurva)
    else:
        Q = self
        R = self if n & 1 == 1 else Ideal(self.kurva)

        i = 2
        while i <= n:
            Q = Q + Q
            if n & i == i:
                R = Q + R
                i = i << 1
            return R

def __rmul__(self, n):
    return self * n

def __list__(self):
    return [self.x, self.y]

```

```

def __eq__(self, other):
    if type(other) is Ideal:
        return False

    return (self.x, self.y)
           == (other.x, other.y)

def __ne__(self, other):
    return not self == other

def __getitem__(self, index):
    return [self.x, self.y][index]

class Ideal(Titik):
    def __init__(self, kurva):
        self.kurva = kurva

    def __neg__(self):
        return self

    def __str__(self):
        return "Ideal"

    def __add__(self, Q):
        if self.kurva != Q.kurva:
            raise Exception("Tidak bisa menambahkan
                             titik dengan kurva yang
                             berbeda!")

        return Q

    def __mul__(self, n):
        if not isinstance(n, int):
            raise Exception("Tidak bisa selain
                             integer untuk
                             multiplikasi skalar")

        else:
            return self

```

```
def __eq__(self, other):
    return type(other) is Ideal
```

2. Potongan kode mod.py

```
from euclidean import *
from numbertype import *
@memoize
def IntegersModP(p):
    class IntegerModP(object):
        def __init__(self, n):
            try:
                self.n = int(n) % IntegerModP.p
            except:
                raise TypeError("Tidak bisa input
                                %s ke %s dalam
                                __init__" %
                                (type(n).__name__,
                                 type(self).__name__))

        self.field = IntegerModP
    @typecheck
    def __add__(self, other):
        return IntegerModP(self.n + other.n)

    @typecheck
    def __sub__(self, other):
        return IntegerModP(self.n - other.n)

    @typecheck
    def __mul__(self, other):
        return IntegerModP(self.n * other.n)

    def __neg__(self):
        return IntegerModP(-self.n)

    @typecheck
    def __eq__(self, other):
        return isinstance(other, IntegerModP)
```

```
and self.n == other.n
```

```
@typecheck
```

```
def __ne__(self, other):  
    return isinstance(other, IntegerModP)  
        is False or self.n != other.n
```

```
@typecheck
```

```
def __divmod__(self, divisor):  
    q,r = divmod(self.n, divisor.n)  
    return (IntegerModP(q), IntegerModP(r))
```

```
def inverse(self):  
    x,y,d = extendedEuclideanAlgorithm(self.n,  
        self.p)
```

```
    if d != 1:  
        raise Exception("Error: p tidak prima  
            dalam %s!" % (self.__name__))
```

```
    return IntegerModP(x)
```

```
def __abs__(self):  
    return abs(self.n)
```

```
def __str__(self):  
    return str(self.n)
```

```
def __repr__(self):  
    return '%d (mod %d)' % (self.n, self.p)
```

```
def __int__(self):  
    return self.n
```

```
IntegerModP.p = p
```

```
IntegerModP.__name__ = 'Z/%d' % (p)
```

```
IntegerModP.englishName = 'IntegersMod%d' % (p)
```

```
return IntegerModP
```

3. Potongan kode euclidean.py

```
def gcd(a, b):
    if abs(a) < abs(b):
        return gcd(b, a)

    while abs(b) > 0:
        _, r = divmod(a,b)
        a,b = b,r

    return a

def extendedEuclideanAlgorithm(a, b):
    if abs(b) > abs(a):
        (x,y,d) = extendedEuclideanAlgorithm(b, a)
        return (y,x,d)

    if abs(b) == 0:
        return (1, 0, a)

    x1, x2, y1, y2 = 0, 1, 1, 0
    while abs(b) > 0:
        q, r = divmod(a,b)
        x = x2 - q*x1
        y = y2 - q*y1
        a, b, x2, x1, y2, y1 = b, r, x1, x, y1, y

    return (x2, y2, a)
```

4. Potongan kode numbertype.py

```
def memoize(f):
    cache = {}

    def memoizedFunction(*args, **kwargs):
        argTuple = args + tuple(kwargs)
        if argTuple not in cache:
            cache[argTuple] = f(*args, **kwargs)
        return cache[argTuple]
    memoizedFunction.cache = cache
    return memoizedFunction
```

```

def typecheck(f):
    def newF(self, other):
        if (hasattr(other.__class__,
                    'operatorPrecedence') and
            other.__class__.operatorPrecedence
            > self.__class__.operatorPrecedence):
        return NotImplemented

    if type(self) is not type(other):
    try:
        other = self.__class__(other)
    except TypeError:
        message = 'tidak bisa typecast
                  %s dari type %s
                  menuju type %s dalam fungsi %s'
        raise TypeError(message % (other, type(other)
                                   ).__name__, type(self).__name__,
                             f.__name__))
    except Exception as e:
        message = 'Type error dala, argumen %r, %r
                  untuk fungsi %s. alasan:%s'
        raise TypeError(message %
                        (self, other, f.__name__,
                         type(other).__name__,
                         type(self).__name__, e))

    return f(self, other)
    return newF

```

4. Potongan kode semuaTitik.py

```

from kurvaeliptik import *
def semuaTitik(kurva, a, b, besar, field):
    xs = []
    ys = []
    points = []
    points.append(Ideal(kurva))
    for i in range(besar):

```



```

for j in range(besar):
    hasilb = (i**3 + a * i + b) % besar
    hasila = (j**2) % besar
    if hasila == hasilb:
        xs.append((field(i)))
        ys.append((field(j)))
        for x, y in zip(xs,ys):
            points.append((Titik(kurva,x,y)))
return points

```

5. Potongan kode dan hasil SAGE interaktif dari Simulasi membuat kurva eliptik

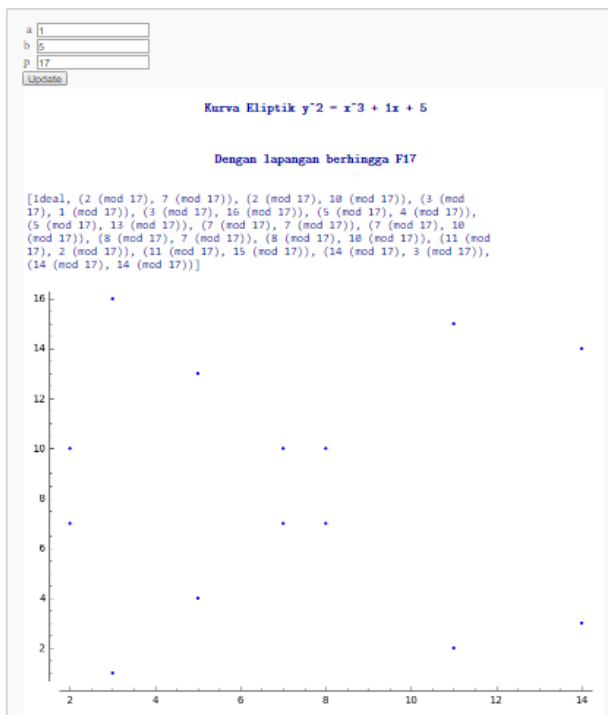
```

%hide
html('<h1>MEMBUAT SELURUH TITIK KURVA
ELIPTIK</h1>')
html('<h1>$y^2 \mod p =x^3+ax+b \mod p$</h1>')
@interact
def seluruhtitik(a = input_box(1, label='a',width=20),
                b = input_box(5, label='b ',width=20),
                p = input_box(17, label='p ',width=20),
auto_update=False):
    F = IntegersModP(p)
    E = KurvaEliptik(F(a), F(b))
    all_points = semuaTitik(E, a, b, p, F)
    show(E)
    show('Dengan lapangan berhingga F%s' %p)
    print all_points
    C = EllipticCurve(GF(p),[a,b])
    show(C.plot())

```

MEMBUAT SELURUH TITIK KURVA ELIPTIK

$$y^2 \bmod p = x^3 + ax + b \bmod p$$



6. Potongan kode dan hasil SAGE interaktif dari Simulasi penambahan dua titik

```
%hide
```

```
html('<h1>PENAMBAHAN DUA TITIK</h1>')
```

```
html('<h2>$P(x,y) + Q(x,y)$</h2>')
```

```
@interact
```

```
def tambahtitik(a = input_box(1, label='a',width=20),
```

```
    b = input_box(5, label='b ',width=20),
```

```
    p = input_box(17, label='p ',width=20),
```

```
    Px = input_box(2, label='Px',width=20),
```

```

Py = input_box(7, label='Py',width=20),
Qx = input_box(3, label='Qx',width=20),
Qy = input_box(16, label='Qy',width=20),
auto_update=False):
    F = IntegersModP(p)
    E = KurvaEliptik(F(a), F(b))
    P = Titik(E,F(Px),F(Py))
    Q = Titik(E,F(Qx),F(Qy))
    print 'P = %s' %P
    print 'Q = %s' %Q
    print 'P + Q = %s' %(P+Q)

```

PENAMBAHAN DUA TITIK

$$P(x, y) + Q(x, y)$$

a
 b
 p
 Px
 Py
 Qx
 Qy

P = (2 (mod 17), 7 (mod 17))
 Q = (3 (mod 17), 16 (mod 17))
 P + Q = (8 (mod 17), 7 (mod 17))

7. Potongan kode dan hasil SAGE interaktif dari Simulasi Pertukaran kunci Diffie Hellman

%hide

html('<h1>DIFFIE-HELLMAN KEY EXCHANGE</h1>')

html('<h2>\$y^2 \mod 7211 = x^3 + x + 7206 \mod p\$</h2>')

html('<h2>\$G(3,5)\$</h2>')

@interact

def diffie(A = input_box(3123, label='Kunci Privat A',width=20),

B = input_box(433, label='Kunci Privat B',width=20),

auto_update=False):

```

F = IntegersModP(7211)
E = KurvaEliptik(F(1), F(7206))
G = Titik(E,F(3),F(5))
PA = int(A)*G
PB = int(B)*G
SKA = int(A)*PB
SKB = int(B)*PA
print 'G = %s' %G
print 'PA = Kunci Privat A * G = %s ' %PA
print 'PB = Kunci Privat B * G = %s ' %PB
print 'Kemudian mereka saling bertukar kunci Publik'
print 'Hal ini dilakukan untuk mendapatkan Kunci Rahasia
Bersama SK'
print 'SKA = Kunci Privat A * PB = %s ' %SKA
print 'SKB = Kunci Privat B * PA = %s ' %SKB
print 'Jelas Bahwa SKA = SKB'

```

DIFFIE-HELLMAN KEY EXCHANGE

$$y^2 \mod 7211 = x^3 + x + 7206 \mod p$$

$G(3, 5)$

G = (3 (mod 7211), 5 (mod 7211))

PA = Kunci Privat A * G = (6432 (mod 7211), 4779 (mod 7211))

PB = Kunci Privat B * G = (5104 (mod 7211), 399 (mod 7211))

Kemudian mereka saling bertukar kunci Publik

Hal ini dilakukan untuk mendapatkan Kunci Rahasia Bersama SK

SKA = Kunci Privat A * PB = (2456 (mod 7211), 15 (mod 7211))

SKB = Kunci Privat B * PA = (2456 (mod 7211), 15 (mod 7211))

Jelas Bahwa SKA = SKB

8. Potongan kode dan hasil SAGE interaktif dari Simulasi Enkripsi-Dekripsi

%hide

html('<h1>ENKRIPSI-DEKRIPSI</h1>')

html('<h2>\$y^2 \mod 7211 =x^3+x+7206 \mod 7211\$</h2>')

html('<h3>\$G(3,5)\$</h3>')

html('<h3>Lena 100</h3>')

```

lena100 = 'data/lena100.jpg'
doni100 = 'data/doni100.jpg'
nicole100 = 'data/nicole100.jpg'
@interact
def en(img = input_box(lena100, label='Citra Input',width=20),
      A = input_box(52, label='Kunci Privat A ',width=20),
      B = input_box(21, label='Kunci Privat B ',width=20),
      auto_update=False):
    start = time.time()
    F = IntegersModP(7211)
    E = KurvaElptik(F(1), F(7206))
    G = Titik(E,F(3),F(5))
    all_points = semuaTitik(E, 1, 7206, 7211, F)
    PA = int(A)*G
    PB = int(B)*G
    SKA = int(A)*PB
    SKB = int(B)*PA
    im = cv2.imread(img)
    lenagray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    x = lenagray.tolist()
    pixelto = []
    pixelto_gen = dict.fromkeys(range(0, 256))
    pixelto_gen1 = dict.fromkeys(range(0,(len(all_points)%256)))

    for i in range(28):
        pixelto.append(pixelto_gen)
    pixelto.append(pixelto_gen1)

    mapping = []
    for i, isi in enumerate(pixelto):
        mapping.append({})
        for j, konten in enumerate(isi):
            mapping[i][konten] = all_points[(i*256)+int(konten)]

```

```

mapping_2 = []
for i, isi in enumerate(pixelto):
    mapping_2.append({})
    for j, konten in enumerate(isi):
        mapping_2[i][konten] = all_points[(i*256)+int(konten)]

del mapping_2[-1]

x_mapping = [[]]
for i, isi in enumerate(x):
    if i>0:
        x_mapping.append([])
        for j, konten in enumerate(isi):
            rnd = random.choice(mapping_2)
            x_mapping[i].append(rnd[konten])

encrypted_point = [[]]
for i, isi in enumerate(x_mapping):
    if i>0:
        encrypted_point.append([])
        for j, konten in enumerate(isi):
            addition = konten + SKA
            encrypted_point[i].append(addition)

encrypted_image = [[]]
for i, isi in enumerate(encrypted_point):
    if i>0:
        encrypted_image.append([])
        for j, konten in enumerate(isi):
            for k in mapping:
                for l in k:
                    if konten == k[l]:

```

```

        encrypted_image[i].append(l)

decrypted_point = [[]]
for i, isi in enumerate(encrypted_point):
    if i>0:
        decrypted_point.append([])
    for j, konten in enumerate(isi):
        subtraction = konten - SKB
        decrypted_point[i].append(subtraction)

decrypted_image = [[]]
for i, isi in enumerate(decrypted_point):
    if i>0:
        decrypted_image.append([])
    for j, konten in enumerate(isi):
        for k in mapping:
            for l in k:
                if konten == k[l]:
                    decrypted_image[i].append(l)

cv2.imwrite('enkripsilena100.jpg',np.array(encrypted_image))
cv2.imwrite('dekripsilena100.jpg',np.array(decrypted_image))
end = time.time()
print 'elapsed time %s s'%(end - start)

```

ENKRIPSI



$$y^2 \bmod 7211 = x^3 + x + 7206 \bmod 7211$$

$G(3,5)$

Cameraman

Citra Input	<input type="text" value="data/mandrill50.jpg"/>
Kunci Privat A	<input type="text" value="52"/>
Kunci Privat B	<input type="text" value="21"/>
<input type="button" value="Update"/>	

enkripsi 89.3278670311 s
dekripsi 23.9079711437 s

BIODATA PENULIS



Penulis memiliki nama lengkap Doni Rubiagatra, lahir di Surabaya pada tanggal 16 September 1993. Penulis berasal dari Kota Surabaya, bertempat tinggal di Jalan Ikan Sepat 4/28 Surabaya. Pendidikan formal yang pernah ditempuh yaitu SD Hang Tuah 2 Surabaya, SMP Negeri 2 Surabaya, dan SMAN 9 Surabaya. Kemudian, penulis melanjutkan studi di jurusan Matematika ITS, dengan bidang minat ilmu komputer. Dalam bidang minat ini penulis mulai mengenal bahasa pemrograman diantaranya adalah C, C++, Java, PHP-MySQL, dan MATLAB, Python, PostgreSQL, R, SageMath. Semasa menempuh jenjang pendidikan S-1, penulis juga aktif dalam kegiatan non-akademis di organisasi kemahasiswaan diantaranya adalah HIMATIKA ITS dan menjadi Ketua BEM FMIPA ITS pada periode 2013/2014. Selama penulisan Tugas Akhir ini Penulis tidak lepas dari kekurangan, untuk itu penulis mengharapkan kritik, saran, dan pertanyaan mengenai Tugas Akhir ini yang dapat dikirimkan melalui *e-mail* ke rubiagatra@gmail.com ataupun nomor telepon 081615462286